OIPF

Release 2 Specification

HTTP Adaptive Streaming

[V2.0] – [2010-09-07]

Open IPTV Forum

Open IPTV Forum

Postal address

Open IPTV Forum support office address
650 Route des Lucioles – Sophia Antipolis
Valbonne – FRANCE
Tel.: +33 4 92 94 43 83
Fax: +33 4 92 38 52 90

Internet
http://www.oipf.tv

Disclaimer

The Open IPTV Forum members accept no liability whatsoever for any use of this document.

This specification provides multiple options for some features. The Open IPTV Forum Profiles specification complements the Release 2 specifications by defining the Open IPTV Forum implementation and deployment profiles. Any implementation based on Open IPTV Forum specifications that does not follow the Profiles specification cannot claim Open IPTV Forum compliance.

Copyright Notification

Contents

Tables

Figures

Foreword

This Technical Specification (TS) has been produced by the Open IPTV Forum.

This specification provides multiple options for some features. The Open IPTV Forum Profiles specification will complement the Release 2 specifications by defining the Open IPTV Forum implementation and deployment profiles. Any implementation based on Open IPTV Forum specifications that does not follow the Profiles specification cannot claim Open IPTV Forum compliance.

Introduction

This specification defines the usage of and, where necessary, extensions to the technologies defined in [TS26234] and [TS26244] to enable HTTP based Adaptive Streaming for Release 2 Open IPTV Forum compliant services and devices.

In the case of HTTP Adaptive Streaming, a service provides a Content item in multiple bitrates in a way that enables a terminal to adapt to (for example) variations in the available bandwidth by seamlessly switching from one version to another, at a higher or lower bitrate, while receiving and playing the Content. This is achieved by encoding a Content item in alternative Representations of different bitrates and segmenting these Representations into temporally aligned and independently encoded Segments. This results in a matrix of Segments as depicted in Figure 1.

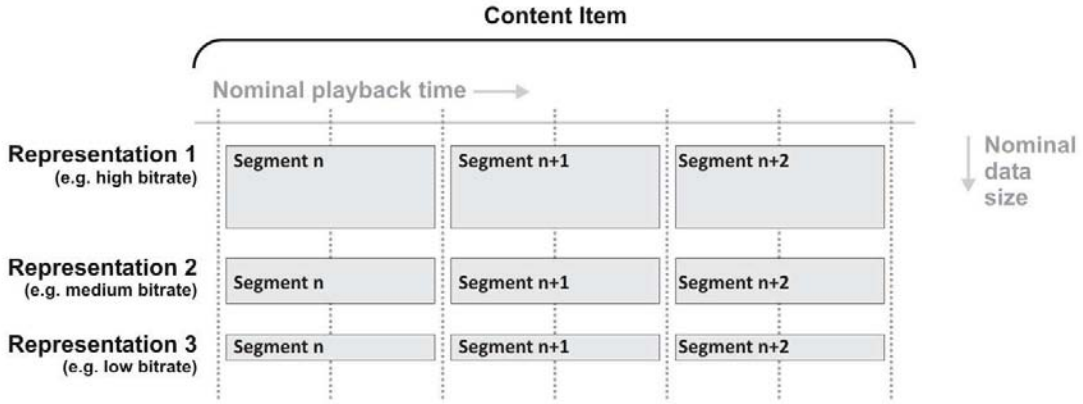

Figure 1: Content Segmentation for HTTP Adaptive Streaming

The Segments are offered for HTTP download from a URL that is unique per Segment. After completion of the download (and playback) of a certain Segment of a certain Representation, a terminal may switch to an alternate Representation simply by downloading (and playing) the next Segment of a different Representation. This requires the terminal to have a description of the available Representations and Segments and the URLs from which to download the Segments. This description is provided as a separate resource: the Media Presentation Description (MPD). The MPD is described in section 3.

The media data in a Segment is formatted in compliance with the media formats as defined in [OIPF_MEDIA2]. However, in the context of HTTP Adaptive Streaming, additional requirements are put on the usage of these formats, especially regarding the systems layers. This “profile” is specified in section 4.

Similarly, the retrieval mechanisms of Segments are in compliance with section 5.3.2.2 of [OIPF_PROT2], with the usage in the context of HTTP Adaptive Streaming as defined in this document.

A Representation may be made up of multiple components, for example audio, video and subtitle components. A partial Representation may only contain some of these components and a terminal may need to download (and play) multiple partial Representations to build up a complete Representation, with the appropriate components according to the preferences and wishes of the user. Appendix C has a more detailed description on the use of partial Representations.

1 References

1.1 Normative References

1.1.1 Standard References

[RFC2119]	IETF RFC 2119 (1997-03), “Key words for use in RFCs to Indicate Requirement Levels”.
[TS101154]	ETSI TS 101 154 V1.9.1 (2009-09), “Digital Video Broadcasting (DVB); Specification for the use of Video and Audio Coding in Broadcasting Applications based on the MPEG-2 Transport Stream”.
[MPEG2TS]	ISO/IEC 13818-1:2000/Amd.3:2004, “Generic coding of moving pictures and associated audio information: Systems”.
[ISOFF]	ISO/IEC 14496-12:2005, “Information Technology - Coding of Audio-Visual Objects - Part 12: ISO Base Media file format”, International Standards Organization.
[OMARLIN]	Marlin Developer Community, “OMArlin Specification”, Version 1.0.1, July 2008.
[MRL FF]	Marlin Developer Community, "Marlin - File Formats Specification", Version 1.1, July 2008, and latest version of “Marlin Errata: Marlin - File Formats Specification V1.1”.
[TS26234]	3GPP TS 26.234, Transparent end-to-end Packet-switched Streaming Service (PSS) Protocols and codecs (Release 9)
[TS26244]	3GPP TS 26.244, Transparent end-to-end packet switched streaming service (PSS), 3GPP file format (3GP) (Release 9)

1.1.2 Open IPTV Forum References

[OIPF_SERV2]	Open IPTV Forum, “Services and Functions for Release 2”, V1.0, October 2008.
[OIPF_REQS2]	Open IPTV Forum, “Service and Platform Requirements”, V2.0, December 2008.
[OIPF_ARCH2]	Open IPTV Forum, “Functional Architecture – V2.0”, September 2009.
[OIPF_MEDIA2]	Open IPTV Forum, “Release 2 Specification, Volume 2 - Media Formats”, V2.0, September 2010.
[OIPF_HAS2]	Open IPTV Forum, “Release 2 Specification, Volume 2a – HTTP Adaptive Streaming”, V2.0, September 2010.
[OIPF_META2]	Open IPTV Forum, “Release 2 Specification, Volume 3 - Content Metadata”, V2.0, September 2010.
[OIPF_PROT2]	Open IPTV Forum, “Release 2 Specification, Volume 4 – Protocols”, V2.0, September 2010.
[OIPF_PROT2_EX]	Open IPTV Forum, “Release 2 Specification, Volume 4a – Examples of Protocol Sequences”, V2.0, September 2010.
[OIPF_DAE2]	Open IPTV Forum, “Release 2 Specification, Volume 5 - Declarative Application Environment”, V2.0, September 2010.
[OIPF_PAE2]	Open IPTV Forum, “Release 2 Specification, Volume 6 - Procedural Application Environment”, V2.0, September 2010.
[OIPF_CSP2]	Open IPTV Forum, “Release 2 Specification, Volume 7 - Authentication, Content Protection and Service Protection”, V2.0, September 2010.

2 Conventions and Terminology

2.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Introduction”, are normative, unless they are explicitly indicated to be informative.

2.2 Terminology

2.2.1 Definitions

In addition to the definitions provided in Volume 1, the following definitions are used in this Volume.

Term	Definition
Content	An instance of audio, video, audio-video information, or data. (from Volume 1). A Content item may consist of several Components.
Component	An element of a Content item, for example an audio or subtitle stream in a particular language or a video stream from a particular camera view.
Component Stream	A bit stream that is the result of encoding a Component with a certain codec and certain codec parameters (e.g. bitrate, resolution).
Content Resource	A Content item that is provided in multiple Representations (e.g. multiple qualities, bitrates, camera views, etc.) to enable adaptive streaming of that Content item. Service Discovery procedures refer to a Content Resource. A Content Resource consists of one or more time-sequential Periods.
Period	A temporal section of a Content Resource.
Representation	A version of a Content Resource within a Period. Representations may differ in the included Components and the included Component Streams.
Segment	A temporal section of a Representation in a specific systems layer format (either MPEG-2TS or MP4), referred to via a unique URL

2.2.2 Abbreviations

In addition to the Abbreviations provided in Volume 1, the following abbreviations are used in this Volume.

Acronym	Explanation
3GPP	ETSI 3rd Generation Partnership Project
AAC	Advanced Audio Coding
AAC LC	AAC Low Complexity
ATSC	Advanced Television Systems Committee
BBTS	Broadband Transport Stream
DCF	DRM Content Format
DRM	Digital Rights Management
DVB	Digital Video Broadcasting
DVB-SI	DVB Service Information
ECM	Entitlement Control Message
ETSI	European Telecommunications Standards Institute
GOP	Group Of Pictures
IPMP	Intellectual Property Management and Protection
IV	Initialisation Vector
JPEG	Joint Photographic Experts Group
MP4	MP4 File Format
MPD	Media Presentation Description
MPEG	Moving Pictures Expert Group
nPVR	Network Personal Video Recorder
NTP	Network Time Protocol
OMA	Open Mobile Alliance
PAT	Program Association Table
PDCF	Packetised DRM Content Format
PF	Protected Format
PID	Packet Identifier
PMT	Program Map Table
RAP	Random Access Point

3 Media Presentation

3.1 Media Presentation Description

The Media Presentation Description (MPD) SHALL be as specified in [TS26234] section 12.2, with the following extensions and additional requirements:

- The MPD SHALL be an XML file that SHALL validate against the schema in Appendix A.
  Note that the XML schema in Appendix A imports the schema specified in [TS26234]. This means that an MPD that does not use any of the OIPF specific extensions will validate against both the schema defined in [TS26234] as well as Appendix A.

- A <Representation> element may carry the @group-attribute set to a non-zero value. In this case the attribute indicates that the <Representation> element is not necessarily a complete Representation, but consists of one or more individual Components (video, audio, subtitle, etc.) which may be downloaded and provided to the terminal in addition to content being downloaded from other <Representation> elements. In this case the <Representation> element SHALL contain one or more <Component> elements, as specified in section 3.1.1, one for each Component contained in the <Representation>. Note that it is the responsibility of the application on the terminal to select the desired Components and to initialize the terminal accordingly. Appendix C contains an informative description of how this can be done. The value of the @group-attribute SHALL be the same for Representations that contain at least one same Component. Two Representations with completely different Components (e.g. audio at two different languages) SHALL have different values for the @group attribute.

An example instance of the OIPF compliant MPD with the constraints from section 3.2 is depicted in Figure 2.

3.1.1 Component Element

Element/ Attribute	Description	Optionality
Component	This element contains a description of a Component.	
@id	Specifies the system-layer specific identifier of the elementary stream of this Component. The value SHALL be equal to the PID of the TS packets that carry the Component Stream of the Component, in case the system layer is MPEG-2 TS. The value SHALL be equal to the track ID of the track that carries the Component Stream of the Component, in case the system layer is MP4.	O
@type	Specifies the Component type. Valid values include “Video”, “Audio” and “Subtitle” to specify the corresponding Component types defined in [OIPF_DAE2], section 7.14.5.1.1.	M
@lang	Specifies an ISO 639 language code for audio and subtitles stream (see [OIPF_DAE2], section 7.14.5). Note that this attribute indicates the language of a specific Component, hence only a single language code is needed. This is different to the usage of the @lang attribute of the <Representation> element in the MPD, which may be used to indicate the list of languages used in the Representation.	O
@description	The value of this attribute SHALL be a user readable description of the Component. This description may be used by the terminal in its user interface to allow a user to select the desired Components, e.g. select from different camera views in case of a video stream.	O
@audioChannels	Specifies the audio channels for an audio stream. (e.g. 2 for stereo, 5 for 5.1, 7 for 7.1 - see [OIPF_DAE2], section 7.14.5). This attribute SHALL only be present when the value of the @type attribute is "Audio".	O
@impaired	When set to “true”, specify that the stream in this Component is an audio description for visually impaired or subtitles for hearing impaired. This attribute SHALL only be present when the value of the @type attribute is "Audio" or "Subtitle".	O
@adMix	When set to “true”, specifies that the Audio stream in this Component must be mixed with (one of the) the main audio stream(s), for which this attribute is absent or set to “false”. This attribute SHALL only be present when the value of the @type attribute is "Audio".	O

Table 1: Component Element and Attributes

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009"
    xmlns:oipf="urn:oipf:iptv:has:2010"
    minBufferTime="PT10S"
    xsi:schemaLocation=
        "urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009      3GPP-MPD-009.xsd
        urn:oipf:iptv:has:2010                              OIPF-MPD-009.xsd"
    >
    <Period start="PT0S" segmentAlignmentFlag="true"  bitStreamSwitchingFlag="true">
        <SegmentInfoDefault
            sourceUrlTemplatePeriod="http://www.aService.com/aMovie/$RepresentationID$/Seg$Index$.3gs"
            duration="PT2S">
        </SegmentInfoDefault>

        <Representation  bandwidth="5000000" mimeType="video/mp4"
            startWithRAP="true" group="1" >
            <SegmentInfo baseURL="http://www.aService.com/aMovie/HQ/" >
                <InitialisationSegmentURL sourceURL="http://www.aService.com/aMovie/Init.mp4"/>
                <Url sourceURL="Seg10.3gs"/>
                <Url sourceURL="Seg11.3gs"/>
                <Url sourceURL="Seg12.3gs"/>
                <!--  Media Segments in high quality available from
                      http://www.aService.com/aMovie/HQ/SegXX.3gs -->
            </SegmentInfo>
            <oipf:Components>
                <oipf:Component type="video" id="1" description="Video"/>
                <oipf:Component type="audio" id="2" lang="en" description="Audio-En"/>
            </oipf:Components>
        </Representation>

        <Representation  bandwidth="2500000" mimeType="video/mp4"
            startWithRAP="true" group="1">
            <SegmentInfo>
                <InitialisationSegmentURL sourceURL="http://www.aService.com/aMovie/Init.mp4"/>
                <Url sourceURL="http://www.aService.com/aMovie/LQ/Seg10.3gs"/>
                <Url sourceURL="http://www.aService.com/aMovie/LQ/Seg11.3gs"/>
                <Url sourceURL="http://www.aService.com/aMovie/LQ/Seg12.3gs"/>
                <!--  Media Segments in low quality  available from
                      http://www.aService.com/aMovie/LQ/SegXX.3gs -->
            </SegmentInfo>
            <oipf:Components>
                <oipf:Component type="video" id="1" description="Video"/>
                <oipf:Component type="audio" id="2" lang="en" description="Audio-En"/>
            </oipf:Components>
        </Representation>

        <Representation   bandwidth="125000" mimeType="video/mp4"
            startWithRAP="true" group="2">
            <SegmentInfo>
                <InitialisationSegmentURL sourceURL="http://www.aService.com/aMovie/Init.mp4"/>
                <UrlTemplate startIndex="10" endIndex="12" id="FR"/>
                <!--  Media Segments with French audio available from
                      http://www.aService.com/aMovie/FR/SegXX.3gs -->
            </SegmentInfo>
            <oipf:Components>
                <oipf:Component type="audio" id="3" lang="fr" description="Audio-Fr"/>
            </oipf:Components>
        </Representation>
    </Period>
</MPD>
```

Figure 2: Example of the MPD

3.2 Segmentation Constraints

The OITF SHALL support Segments as specified in [TS26234] with the following constraints:

- Each Segment SHALL start with a random access point (RAP) and the @startWithRAP attribute SHALL be present and set to 'true' in all <Representation> elements in the MPD.
- Byte Ranges SHALL NOT be used as a mechanism for identifying Segments. As a consequence the elements <InitialisationSegmentURL> and <Url> SHALL NOT include the optional attribute @range. Note that this does not preclude the use of HTTP requests with byte ranges to retrieve parts of a Segment.
- To enable seamless switching:
  - Different Component Streams of the same Component SHALL be encoded in the same media format but MAY be different in the profile of that format. Section 4 in this document references [OIPF_MEDIA2] for the media formats, which specifies (profiles of) media formats for various media types. So if for example a Representation contains a Component Stream of a certain video Component that is encoded using H.264/AVC using the HD profile, then all Representations that have a Component Stream of that Component must use H.264/AVC but may use different configurations of H.264/AVC within the HD profile or SD Profile.
  - Segments of Representations with the same value for the @group attribute SHALL be time aligned. The attributes 'segmentAlignmentFlag' and 'bitstreamSwitchingFlag' SHALL be present and set to 'true' in all 'Period' elements in the MPD.

    For the set of Representations that have the same value for the @group attribute, the signaled Segment durations SHALL:

    - either be equal for all Representations in the set,
    - or equal for all Representations in the set without a <TrickMode> element and a multiple of this value for the Representations in the set with a <TrickMode> element. In this case there SHALL be at least one Representation in the set for which the <TrickMode> element is absent.

    Terminals are RECOMMENDED to select Representations with a <TrickMode> element in case of trickplay and to select Representations without a <TrickMode> element for play at normal speed. Terminals MAY select any Representation for both trickplay and normal play, regardless of the presence of the <TrickPlay> element and differences in duration of the Segments in a group.

    This will enable larger Segments for dedicated trick Representations which MAY be composed of intra-frames only with a fixed interval and therefore avoid that the number of Segment downloads per second is excessive during trick modes. NOTE: A non-time-aligned trick play Representation makes switching between it and the media Representation more difficult to achieve seamlessly, or less accurate for an OITF that does not perform extra seek processing.
- If two <InitialisationSegmentURL>–elements have the same value in the sourceURL attribute, then the referenced init-data SHALL be the same. Consequently the terminal does not need to download the init-data twice.
- All Representations assigned to a non-zero group SHALL carry an <InitialisationSegmentURL> element with the same value of the @sourceURL attribute. The referenced Initialization Segment SHALL carry the metadata that describes the samples for all Representations assigned to a non-zero group. A client only needs to acquire this overall Initialization Segment once.

Note that if a service chooses to Segment a Content Resource in a way that does not meet these constraints, then the Content Resource might not be supported on all receivers.

3.3 Signaling of Content Protection in the MPD

If Segments are protected, then the corresponding <Representation>-element in the MPD SHALL have a <ContentProtection> child element as specified in [TS26234]. The @schemeIdUri-attribute of the <ContentProtection>-element SHALL be set equal to the DRMSystemID as specified in [META]. For example, for Marlin, the DRMSystemID and @schemeIDUri-attribute value is “urn:dvb:casystemid:19188”. [TS26234] allows a <SchemeInformation>-element to be located in the <ContentProtection>-element, however usage of this feature is not defined in this specification (i.e. if it is present, it may be ignored).

3.4 Media Presentation Description Updates

Streaming of live Content SHALL be done following the rules described in [TS26234]: the MPD may be updated periodically at the interval described in the MPD, and successive versions of the MPD are guaranteed to be identical in the description of Segments that are already in the past. The synchronization of terminals and the live streaming server is addressed by external protocols such as NTP or equivalent.

If service provider provides nPVR functionality to support a timeshift service using network storage, the following applies:

- When the Segments of the live Content are stored on the nPVR server, which would occur after the timeShiftBufferDepth has passed, the URLs indicating the Segments on the nPVR server SHOULD be provided to the OITF to enable it to access these Segments at their new location by the MPD update mechanism [TS26234].

- The updated MPD SHOULD contain new URLs of the Segments on the nPVR server; these SHOULD have the same availabilityStartTime as in the original MPD.

4 Adaptive Media Formats

The video, audio and subtitle formats used for HTTP Adaptive Streaming are the same as those defined in [OIPF_MEDIA2]. As in [OIPF_MEDIA2], at the systems layer, two formats for HTTP Adaptive Streaming are defined, namely MPEG-2 Transport Stream and MP4 File Format.

4.1 MPEG-2 Transport Stream Systems Layer

If the Representation@mimeType attribute equals “video/mpeg” or “video/mp2t”, the media of a Representation is encapsulated in MPEG2-Transport Stream packets and the carriage of A/V Content and related information SHALL be in compliance with the [OIPF_MEDIA2] requirements on usage of the MPEG2-TS systems layer format, with the exceptions and additional requirements listed in sections 4.1.1 through 4.1.5.

4.1.1 PID Allocation

- Regardless of the allocation of Component Streams to Representations,
  - Component Streams of the same Component SHALL be carried in transport stream packets that have the same PID (in transport stream packet header) and the same stream_id (in PES packet header).
  - Component Streams of different Components SHALL be carried in transport stream packets that have different PIDs (in transport stream packet header).
  - Some examples:
    - "audio in Spanish" and "audio in English" have different PID
    - "audio in English" and "audio description for impaired in English" have different PID
    - "audio description in English at 64kbps" and "audio description in English at 128kbps" have the same PID
    - "video angle 1 in H.264 at 720x576" and "video angle 1 in H.264 at 320x288" have the same PID.
- When the Segments of a Representation contain MPEG-2 TS packets, the value of the id attribute in each Component element, if present, SHALL be the PID of the Transport Stream packets which carry the Component

4.1.2 Program Specific Information

- For all Representations, the PAT and PMT, either contained in the Initialisation Segments or in the media Segments, SHALL always contain the full list of all elementary streams. This means that Representations with the @group attribute set to zero will have the same PAT/PMT as Representations with the @group attribute set to a non-zero value . It will be responsibility of the application to apply in the terminal the required PID filters for the Components which are effectively being retrieved through the HTTP adaptive protocol.
- If the media Segments do not contain PAT and PMT tables, the Initialisation Segment SHALL be present and declared in the MPD, pointing to a resource containing transport stream packets with at least one PAT and one PMT

4.1.3 Access Unit Signaling

- The random_access_indicator and elementary_stream_prioirty indicator are set as specified in section 4.1.5 and 5.5.5 of [TS101154].
- It is RECOMMENDED that all transport streams packets where a video frame starts carry a non-empty AU_information data field as defined in annex D.2.2 of [TS101154]
- The inclusion of the above signaling SHALL be used in a consistent manner for all Components in all Segments for a Content item.

4.1.4 Media Packaging

- A media Segment SHALL contain the concatenation of one or several contiguous PES packets which are split and encapsulated into TS packets. Media Segments SHALL contain only complete PES packets.
- When packetizing video elementary streams, up to one frame SHALL be included into one PES packet. Larger frames may be fragmented into multiples PES packets. The PES packet where a frame starts SHALL always contain a PTS/DTS header fields in the PES header.
- PTS and DTS values SHALL be time aligned across different Representations.
- There may be a discontinuity of the “continuity counter” in TS packets when changing from one Representation to another. The OITF SHALL expect that there might be a discontinuity on the “continuity counter” when changing from one Representation to another.

4.1.5 Content Protection

[OIPF_MEDIA2] specifies two methods to protect (i.e. encrypt) MPEG-2 transport streams: BBTS and PF.

The following requirements apply if Segments are protected:

- Initialisation Segment and the Media Segments SHALL be formatted such that a file that consists of the Initialisation Segment and an arbitrary selection of Media Segments of the (set of partial) Representation(s), stored in order of their index in the MPD, is an BBTS compliant file or a PF compliant file or both. This MAY be achieved by using the same Crypto-period boundaries and Control Words across different Representations.
- The DRM related metadata (i.e. PMT containing CA descriptors, CAT, EMM streams or ECM streams) in relation to a certain elementary stream SHALL be delivered as part of either the Media Segments that carry the samples of the elementary stream or the Initialisation Segment.
- The DRM related metadata (i.e. ECM stream) of a certain protection system (i.e. Conditional Access system in MPEG2TS terminology) in relation to a certain elementary stream SHALL have the same PID in all Segments in which it is included. Example: the BBTS defined ECM’s for the audio in Spanish is always PID 134, in all Representations where Spanish audio is present, at any bitrate.

4.2 MP4 File Format Systems Layer

If the Representation @mimeType attribute equals “video/mp4”, then the carriage of A/V Content and related information (e.g. subtitles) SHALL be in compliance with the [OIPF_MEDIA2] requirements on usage of the MP4 systems layer format, with the following restrictions:

- For every Representation, a [TS26234] Initialisation Segment SHALL be available.
    - For all Representations, a reference to the Initialisation Segment SHALL be present in a <InitialisationSegmentURL> element in the <Representation> element.
    - An Initialisation Segment SHALL be delivered with MIME type “video/mp4”.
    - Initialisation Segments SHALL be formatted as specified in [TS26234], section 12.4.2.2. For every media stream of the (set of partial) Representation(s), the *moov*-box in the Initialisation Segments SHALL contain a *trak*-box describing the samples of the media streams in compliance with [ISOFF].
- Every Representation SHALL consist of Media Segments that are formatted as specified in [TS26234], section 12.4.2.3.
    - A Media Segment SHALL be delivered with MIME type “video/vnd.3gpp.Segment” as specified in [TS26244]
    - To allow a terminal to seek to any Segment with a certain index and start playback with perfect audio/video synchronization, every *traf*-box of a track that contains audio SHOULD contain a [TS26244] *tfad*-box. The contents of the box SHALL be such that if the terminal starts the playback of the audio samples of a Segment as specified in the box, then the audio and video of the Segment are played in perfect sync.
- The Initialisation Segment and the Media Segments are formatted such that a file that consists of the Initialisation Segment and an arbitrary selection of Media Segments of either any complete Representation (@group attribute equal to zero) or the set of partial Representations (@group attribute unequal to zero), stored in order of the *sequence_number* in their *mfhd*-box (i.e. increasing order and no duplicates), is an [ISOFF] compliant file. (Note that this statement assumes that [ISOFF] allows for ‘gaps’ in the *sequence_number*s of consecutive *moof*-boxes; i.e. the difference in the *sequence_number* of consecutive *moof*-boxes may be larger than one).
- Regardless of the allocation of Components Streams to Representations,
    - Component Streams of the same Component SHALL be carried in track fragments that have the same trackID (in the *tfhd*-box).
    - Component Streams of different Components SHALL be carried in track fragments that have different trackIDs (in the *tfhd*-box).
- If the Segments are protected, the Initialisation Segment and Media Segments SHALL also meet the requirements as specified in section 4.2.1.

An informative appendix on the use of the MP4 file format systems layer is provided in Appendix D.

4.2.1 Content Protection

[OIPF_MEDIA2] specifies three methods to protect (i.e. encrypt) MP4-based file formats: DCF, PDCF and MIPMP. This specification does not specify how to apply the DCF file format in the context of adaptive streaming.

The following requirements apply if Segments are protected:

- Initialisation Segment and the Media Segments SHALL be formatted such that a file that consists of the Initialisation Segment and an arbitrary selection of Media Segments of either any complete Representation (@group attribute equal to zero) or the set of partial Representations (@group attribute unequal to zero), stored in order of the sequence_number in their *mfhd*-box (i.e. increasing order and no duplicates), is either a PDCF [OIPF_MEDIA2] compliant file or a MIPMP [OIPF_MEDIA2] compliant file.

- The DRM related metadata SHALL be delivered as part of the Initialisation Segment:

  - With PDCF format, the DRM related metadata is located in the *moov*-box. In addition, some DRM related metadata could also be contained in a Mutable DRM Info box. If used, the Mutable DRM Info box SHALL be delivered as part of the Initialisation Segment and located after the *moov*-box.

  - With MIPMP format, the DRM related metadata is not located in the *moov*-box but referenced from the *moov*-box as a separate track carrying an Object Descriptor Stream. The samples of the IPMP Object Descriptor Stream SHALL be delivered as part of the Initialisation Segment in a dedicated *mdat*-box located after the *moov*-box.

NOTE: MIPMP uses cipher block chaining mode, whereas PDCF allows cipher block chaining mode or counter mode. When cipher block chaining is used for encryption, Media Segments need to be encrypted independently of each other. Given that this specification requires a Segment to start with a RAP and given that both MIPMP and PDCF require each access unit to start with its own IV and be encrypted separately, no additional requirements are needed to achieve independent encryption of media Segments.

The “Access Unit Format Header”, as defined for the PDCF format allows the generation of samples that are identical to samples that comply with the MIPMP defined method of “Stream Encryption”. This means that a Service Provider may simultaneously address devices that support the MIPMP format and devices that support the PDCF format by providing different Initialisation Segments for the same Media Segments. The following additional constraints to the PDCF encryption method achieve this:

- the PDCF “Encryption Method” is set to AES_128_CBC (cipher block chaining mode)

- "PaddingScheme" is set to RFC_2630 (Padding according to RFC 2630)

- “SelectiveEncryption” is not used.

5 Use Cases (Informative)

5.1 Live Streaming

If the @timeShiftBufferDepth attribute is present in the MPD, it may be used by the terminal to know at any moment which Segments are effectively available for downloading with the current MPD. If this timeshift information is not present in the MPD, the terminal may assume that all Segments described in the MPD which are already in the past are available for downloading.

When contents provider updates the MPD for live streaming, the new MPD should include all available Segments including the Segments included in the previous MPD. If the sum of timeShiftBuffer in the previous MPD and Segment duration in the previous MPD is larger than NOW-availabilityStartTime in the current MPD, the playlist should include the combination of the 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; CheckTime] of the current MPD and the previous MPD.

Periods may be used in the live streaming scenario to appropriately describe successive live events with different encoding or adaptive streaming properties. Timeshift is still possible across the boundaries of such events, provided that the timeshift window is large enough.

5.2 Trick Play

Following the principles included in the 3GPP specification, the basic implementation of trick modes (fast forward, fast rewind, slow motion, slow rewind, pause and resume) is based on the processing of Segments by the terminal software: downloaded Segments may be provided to the decoder at a speed lower or higher than their nominal timeline (the internal timestamps) would mandate, thus producing the desired trick effect on the screen. Under these conditions the timestamps and the internal clock, if any, in the downloaded Segments do not correspond to the real time clock in the decoder, which need to be set appropriately.

Pausing a Media Presentation can be implemented by simply stopping the request of Media Segments or parts thereof. Resuming a Media Presentation can be implemented through sending requests to Media Segments, starting with the next fragment after the last requested fragment. Slow motion and slow rewind can be implemented through controlling the normal stream playout speed at client side. The rest of this section addresses fast forward and fast rewind implementation.

The playback of Segments in fast forward and fast rewind has an immediate effect on the bitrate that is effectively required in the network, because the Segments also need to be downloaded at a faster or at a slower rate than in normal play mode. The terminal should take this into account when doing the bitrate calculations for implementing the adaptive protocol. Dedicated stream(s) may be used to implement efficient trick modes: it is recommended to produce the stream(s) with a lower frame rate, longer Segments or a lower resolution to ensure that the bitrate is kept at a reasonable level even when the Segment is downloaded at a faster rate. The dedicated stream is described as Representation with a <TrickMode> element in the MPD. It is also recommended that if there are dedicated fast forward Representations, the normal Representations do not contain the <TrickMode> element in the MPD

A very low bitrate version of video might be used to implement some trick speeds, even if that Representation was not created with trick modes in mind; note however that in this case it is possible that the terminal would inject a very high frame rate to the decoder (yet at an acceptable bitrate).

For fast rewind trick modes the terminal downloads successive Segments in reverse order, and it also requires that the frames corresponding to the Segment are presented in reverse order with respect to the capturing/encoding order. The feasibility of this process depends on the capability of the decoder and also on the encoding properties of the stream (e.g. it may be easier to implement if the Segment has been encoded using only intra frames)

In order to start trick mode and easily switch between trick and normal play mode at anytime and support for reverse playing, the trick mode streams may be composed of intra frame only with a fixed interval.

5.3 MPEG-2 TS Seeking

To determine the random access point in a media Segment, the client should download and search RAP one by one till the required RAP is found. The ‘random_acess_indicator’ and ‘elementary_stream_priority_indicator’ in adaptation field of the transport stream may be used for locating every RAP.

Appendix A. MPD Schema (Normative)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns="urn:oipf:iptv:has:2010"
      targetNamespace="urn:oipf:iptv:has:2010"
      xmlns:pss="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009"
      >
      <xs:import namespace="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009" schemaLocation="3GPP-MPD-009.xsd" />

      <xs:element  name="Component" type="ComponentType"/>
      <xs:element  name="Components" type="ComponentsType"/>

      <xs:complexType name="ComponentsType">
            <xs:sequence>
                  <xs:element minOccurs="1" maxOccurs="unbounded" ref="Component"/>
            </xs:sequence>
      </xs:complexType>

      <xs:complexType name="ComponentType">
            <xs:attribute name="id" type="xs:string" use="required"/>
            <xs:attribute name="type" type="xs:string" use="required"/>
            <xs:attribute name="lang" type="xs:string" use="optional"/>
            <xs:attribute name="description" type="xs:string" use="optional"/>
            <xs:attribute name="audioChannels" type="xs:unsignedInt" use="optional"/>
            <xs:attribute name="impaired" type="xs:boolean" use="optional"/>
            <xs:attribute name="adMix" type="xs:boolean" use="optional"/>
      </xs:complexType>
</xs:schema>
```

Figure 3: MPD Schema

Appendix B. HTTP Adaptive Streaming Initiation (Normative)

B.1 Initiation from DAE

Of the three methods specified in section 4.7.1 of [OIPF_DAE2] for the launch of unicast streaming of a CoD item, the second and third methods are applicable to HAS content.

For method 2, the “data” property of the CEA-2014 A/V Control object SHALL be set to a URL that points to a MPD.

For method 3, the Content Access Streaming Descriptor SHALL contain <ContentURL> elements that points to an MPD.

If the OITF supports HAS then it SHALL support setting up the HAS stream using the URL provided via either of these two methods.

The MPD SHALL be delivered with the MIME type as specified for the MPD in [TS26234], i.e. “video/vnd.3gpp.mpd”.

To this end, the OITF SHALL fetch the MPD from the URL, after which the MPD SHALL be interpreted and an initial (set of partial) Representation(s) selected. After this playback can be started when the play() method is invoked.

If the MPD is not valid according to the XML Schema and semantics as defined in Annex A, the A/V control object SHALL go to playState 6 (i.e. error), with error value 4 – content corrupt or invalid.

If the HTML 5 video is supported as defined in section 9.3.16 of [DAE2] and the OITF supports HAS then the OITF SHALL support setting the src attribute of a <video> tag to a URL that points to an MPD.

If the OTF supports HAS then it SHALL support creating a Channel object using the `createChannelObject( Integer idType, Integer onid, Integer tsid, Integer sid, Integer sourceID, String ipBroadcastID )` method where the idType argument is ID_IPTV_URI and the ipBroadcastID argument is a URL which points to an MPD for Scheduled Content (live streaming) over HTTP.

B.2 Initiation from PAE

In [OIPF_PAE2], HAS support SHALL be signaled by setting the following System Property:

- **org.oipf.supportsHTTPAdaptiveStreaming**:
  The PAE supports HAS (Yes / Null).

When AG supports HAS, the Player object SHALL accept URL, MediaLocator and DataSource objects pointing to a MPD. The MPD SHALL be delivered with the MIME type as specified for the MPD in [TS26234], i.e. “video/vnd.3gpp.mpd”.

The Player, during the *Realizing* state, SHALL fetch the MPD from the URL, after which the MPD SHALL be interpreted and an initial (set of partial) Representation(s) selected. If all the operations are completed successfully the player moves to the *Realized* state and then to the *Prefetching* and *Prefetched* state. After this, playback can be started (e.g.: invoking the start() method of the Player).

If the MPD is not valid according to the XML Schema and semantics as defined in Annex A, the Player object SHALL move to the *Unrealized* state. In this case a javax.media.ResourceUnavailableEvent will be sent to all subscribed listeners.

If the MIME type specified for the MPD cannot be handled by the AG (i.e.: the AG does not support HAS), the creation of a Player object with URL, MediaLocator and DataSource objects pointing to a MPD will throw a NoPlayerException exception.

Appendix C. Component Management (Informative)

A <Representation> element with the @group attribute set to zero as defined in [TS26234] corresponds to a particular version of the full Content item with all its elements (video, audio, subtitles, etc). If all Representations have the @group attribute set to zero, the different Representations listed in the MPD correspond to full, alternate versions that differ in one or more particular aspects (bitrate, language, spatial resolution, etc). This means that the terminal needs at every moment to download and present Segments of only one Representation. While this provides a quite simple and straightforward model it has an important lack of flexibility in the following sense: if there are many alternatives for a particular Component (e.g. audio in different languages) and there are also a number of different bitrate alternatives, all combinations should be available at the server and consequently some media data is redundantly stored.

For example, if a service provides 2 audio languages and the video in 2 bitrate levels, then it would need to provide 4 different Representations; however, there will be groups of 2 Representations which share exactly the same bulky video (they only differ in audio). This causes an important waste of storage space in the server. Even if the server can be optimized with respect to this (e.g. to build the Segments in real time from the elementary streams stored separately in its disks), this cannot be done in standard the HTTP caches.

In order to solve this problem, [TS26234] includes the concept of partial Representations in the MPD though the @group attribute. When this attribute has a value different from 0, the Representation does not include all Components of the Content Resource, but only a subset of them (e.g. “audio in French”). An OIPF terminal needs to be able to identify the Representations that it requires, download their Segments independently and combine them for playback at the terminal side.

In case of the example service above, the server may serve 2 Representations with 2 different bitrate versions of a movie with English audio, and separately it can serve a Representation with just the French audio. This way, all combinations are possible (all bitrates at all languages) but with roughly half the required storage in the server and the HTTP caches compared to when all possible combinations are separately stored as complete Representations. Figure 4 depicts the grouping of Components and Component Streams into Representations for this example.

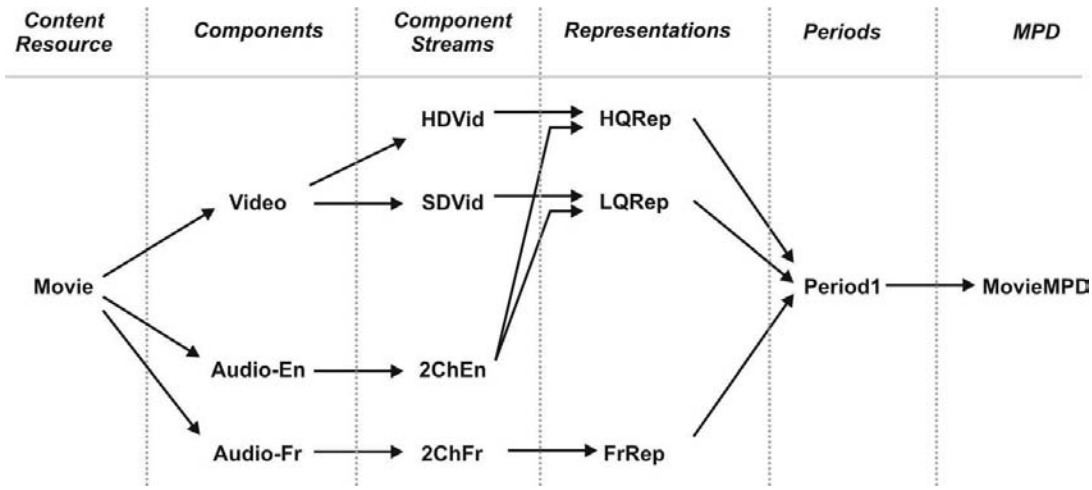

Figure 4: Component management example

In this example the Representations HQRep and LQRep would have the same non-zero value for the @group attribute; FrRep would have a different non-zero value. Additionally both HQRep and LQRep would carry <Component> elements that describe the Video and Audio-En Components; the FrRep would carry a <Component> element for the Audio-Fr Component.

This Component-aware scenario relates to the process for selecting and presenting the desired set of Components. This process may also be applied for Content that is delivered through other mechanisms than the HTTP adaptive streaming protocol described in this document. In the context of OIPF (for example using the DAE “Extensions to video/broadcast for playback of selected Components”), this process operates may utilize information contained in the MPEG2-TS or MP4 metadata. Information contained in the Initialisation Segment may also be used in this process..

The following is an example process for Component selection:

1. Retrieve the MPD. If the MPD includes both, decide if you want to play the partial or non-partial Representations. It is RECOMMENDED to use the partial Representations.
2. In case of a non-partial Representations:
   a) Based on metadata in the MPD (typically the @bandwidth-attribute), select an initial Representation.
   b) If present, retrieve the Initialisation Segment of the Representation.
   c) Retrieve Media Segments of the chosen Representation.
   d) Find the elementary streams in the downloaded Initialisation Segment / Media Segments. Typically select one video and one audio stream. If there are options, select from those.
   e) Setup the "player" to play the selected Component Streams. Play them.
   f) While playing, allow the user to select other/additional Component Streams in the Initialisation Segment / Media Segments.
   g) To switch to a different bitrate, select an alternate non-partial Representation and continue from step 2b.
3. In case of a partial Representations:
   a) Based on the metadata in the MPD (typically the @bandwidth-attribute and the <Component> element) select the initial Representations.
   b) If present, retrieve the Initialisation Segment of the Period.
   c) Retrieve Media Segments of the chosen Representations.
   d) Based on the @id's of the <Components> elements, or using information from the Initialisation Segment, setup the "player" to play the selected Component Streams. Play them.
   e) While playing, allow the user to select other/additional Component Streams..
   If other/additional streams are selected, continue from step 3c.
   f) To switch to a different bitrate of one of the chosen partial Representations, select an alternate partial Representation with the same value for the @group attribute and continue from step 3c.

Note that the Initialisation Segment will always contain the full description of all Component alternatives, so it will be guaranteed that there are no identifiers conflicts between them (e.g. two languages with the same MPEG-2 TS PID or MP4 trackID). The parsing of this Initialisation Segment and the corresponding settings on the terminal to select the appropriate Components is a responsibility of the application (the media player).

Appendix D. Usage of the MP4 File Format (Informative)

D.1 Audio/Video Synchronization

Unlike MPEG-2 TS, the MP4 system layer ([ISOFF]) does not define a system clock or global timestamps that link the various elementary streams to the system clock. Instead, every track has its own independent timeline, specified based on the durations of samples. The decoding time of a sample is calculated by summing up the durations of all samples since the start of the track. The composition time of a sample is either identical to the decoding time, or indicated by an offset to the decoding time.

In the context of adaptive streaming (and especially in case of live streaming), a terminal may want to start playback at any point in the Content without having access to the durations of all samples since the start of the track. Audio/video synchronization would not be a problem if at the start of each Segment, audio and video would always be perfectly aligned. This however is not possible, because video frames and audio frames are typically unequal in duration. Consequently, a Segment that contains an integer number of audio and video frames will not have equal durations of audio and video data.

For example, that a movie consists of an audio and a video elementary stream, where the video is sampled at 25fps and the audio is sampled at 48 kHz and framed using 1024 audio samples per frame. This means that the duration of a video frame is 40 ms and the duration of an audio frame is 21,33 ms. Say also that these elementary streams are delivered using this specification and the MP4 system layer, with the following parameters:

- timescale as specified in the *mvhd*-box: 25 (“ticks” per second)
- timescale as specified in the *mdhd*-box of the video track: 25
- timescale as specified in the *mdhd*-box of the audio track: 48000
- Segment duration as specified in the MPD: 2 seconds

For this case, **Table 2** gives an overview of the allocation of audio and video frames to the first 12 Segments of this movie:

Table 2: Example Audio/Video Synchronization

Segment index	0	1	2	3	4	5	6	7	8	9	10	11	12
Video start time (ticks)	0	50	100	150	200	250	300	350	400	450	500	550	600
Audio start time (ticks)	0,00	50,13	100,27	149,87	200,00	250,13	300,27	349,87	400,00	450,13	500,27	549,87	600,00
#Video frames	50	50	50	50	50	50	50	50	50	50	50	50	50
#Audio frames	94	94	93	94	94	94	93	94	94	94	93	94	94
Video duration (ticks)	50	50	50	50	50	50	50	50	50	50	50	50	50
Audio duration (ticks)	50,13	50,13	49,60	50,13	50,13	50,13	49,60	50,13	50,13	50,13	49,60	50,13	50,13

As can be seen in this example audio and video are perfectly aligned in Segments 0, 4, 8 and 12. However if a terminal seeks to for example Segment 5, then it would need to delay play-out of the audio for 0.13 ticks or 5ms compared to the video to achieve perfect audio/video synchronization.

To signal this to the terminal, [TS26244] specifies the *tfad*-box, which this specification recommends to insert into the audio track. **Figure 5** depicts a close up of the situation at the start of Segment 5 in the above example:

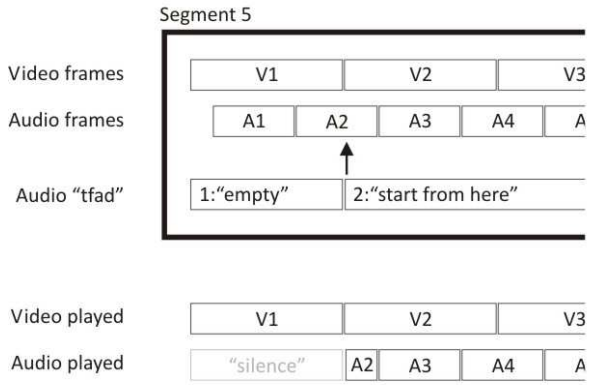

Figure 5: Example *tfad*-box

The *tfad*-box allows adding empty-time into a track at the accuracy of the timescale of the *mvhd*-box, which in this example is 25 and equal to the video. The *tfad*-box also allows specifying to skip certain samples of a track at the timescale of the track, which in this example is 48000. To achieve perfect audio/video sync in this example, Segment 5 may include a *tfad*-box in the audio track with the following contents:

- Entry 1 ("1:empty" in **Figure 5**):
  - Segment_duration= 1
  - media_time= -1 (i.e. "empty" time)
- Entry 2 ("2:start from here" in **Figure 5**):
  - Segment duration=99
  - media_time=1664

A client that starts playing at Segment 5 may use this box to synchronize audio and video, which will result in the playing of the samples as depicted in the bottom half of **Figure 5**. A terminal that continues playing the Content from Segment 4 where it already has synchronized the audio and video tracks and should ignore the *tfad*-box and add the samples of Segment 5 back to back with the samples of Segment 4.

D.2 Partial Representations

Via partial Representations, this specification allows services to offer the various elementary streams of a presentation as separate downloads/streams (see Appendix C). In this case it is required that there is one single Initialisation Segment describing the samples in all Media Segments of all partial Representations and that the concatenation of the Initialisation Segment and the Media Segments is an [ISOFF] compliant file. This section illustrates how such requirement can be met by working out the example of Appendix C in combination with the MP4 system layer.

In this example a service offers a video in 2 bitrates and audio in 2 languages, English and French, where French audio is offered for separate retrieval as a separate partial Representation (see Figure 4). **Figure 6** depicts a potential allocation of movie and track fragments to Segments and Representations for the first few Segments of this example.

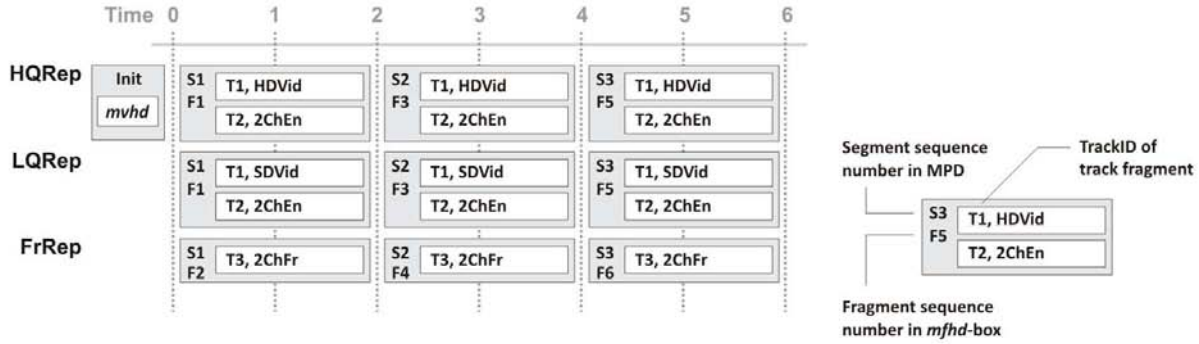

Figure 6: Partial Representation MP4 Example

In the above example, each Segment has a sequence number in the MPD (i.e. the Segment index value) and contains a single movie fragment with a sequence number in the *mfhd*-box. Segments of the Representations "HQRep" and "LQRep" contain samples of both audio (English) and video tracks. Note that in this example the service is required to put each alternate video track on the same TrackID and define a common Init Segment for all partial Representations. Consequently each Component Stream will have its own sample description (in the *trak*-box of track 1) in the common *mvhd*-box.

If a terminal selects to retrieve French audio in combination with the video, then it may retrieve the sequence of Segments as depicted in **Figure 7**.

Figure 7: Partial Representation Retrieval

When stored as depicted (Initialisation Segment first, Media Segments in increasing order of movie fragment sequence number) this is a valid [ISOFF] file that can be played on an existing MP4 player.

Note that the MPD could also include additional non-partial Representations, that reference the same Media Segments as the HQRep and LQRep Representations in this example, and the same (or a different) Initialisation Segment. In this way the same service (and the same HTTP caches!) can be used for terminals that do not support partial Representations.