OIPF
Feature Package

Implementation Guideline for STB-less IPTV

[V1.0] – [2014-05-30]

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 accepts no liability whatsoever for any use of this document.

Copyright Notification

No part may be reproduced except as authorized by written permission.
Any form of reproduction and/or distribution of these works is prohibited.

Copyright © 2014 Open IPTV Forum e.V.

All rights reserved.

Contents

 Copyright 2014 © Open IPTV Forum e.V.

Tables

Figures

 Copyright 2014 © Open IPTV Forum e.V.

Foreword

This document contains Implementation Guideline for STB-less IPTV, part of the 'Additional Features to Support STB-less IPTV' Feature Package.

Introduction

This annex presents an implementation guideline for a solution to deliver IPTV without a set top box from an end-to-end service perspective. It focusses on the interworking between an IPTV App and the TV, and the relation between the CI+ interface and the TV interface.

 Copyright 2014 © Open IPTV Forum e.V.

1 Scope

This document contains Implementation Guideline for STB-less IPTV, part of the ‘Additional Features to Support STB-less IPTV’ Feature Package.

This document provides a normative description of an IPTV implementation based on the OIPF Release 2.3 Solution specifications and certain other specifications, pulling together those specifications and specifying what parts need to be implemented in order to achieve interoperability. Unless stated otherwise, any text included in this document is considered Normative for implementations.

 Copyright 2014 © Open IPTV Forum e.V.

2 References

2.1 Normative References

[CI+1.3]	CI Plus LLP, “CI Plus Specification" V1.3, January 2011
[CI+1.4]	Digital Video Broadcasting, ”Extensions to the CI Plus Specification”, ETSI TS 103 205 v1.1.1, March 2014
[HbbTV1.5]	European Telecommunications Institute (ETSI), ETSI TS 102 796 V1.2.1, “Hybrid Broadcast Broadband TV”, November 2012

2.2 Open IPTV Forum References

[OIPF_DAE2]	Open IPTV Forum, “Release 2 Specification, Volume 5 - Declarative Application Environment” V2.3, January 2014
[OIPF_DAE2-WEB]	Open IPTV Forum, “Release 2 Specification, Volume 5a – Web Standards TV Profile” V2.3, January 2014
[OIPF_PROT2]	Open IPTV Forum, “Release 2 Specification, Volume 4 - Protocols” V2.3, January 2014
[OIPF_HAS2]	Open IPTV Forum, “Release 2 Specification, Volume 2a – HTTP Adaptive Streaming” V2.3, January 2014
[OIPF_MEDIA2]	Open IPTV Forum, “Release 2 Specification, Volume 2 – Media Formats” V2.3, January 2014
[STB-LESS-FP]	Open IPTV Forum, “Feature Package: Additional Features to Support STB-less IPTV”, V1.0, May 2014

 Copyright 2014 © Open IPTV Forum e.V.

3 Conventions, Definitions and Abbreviations

3.1 Conventions

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

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].

3.2 Abbreviations

In addition to the Abbreviations made in Volume 1, the following abbreviations are used in the scope of this volume.

Abbreviation	***Definition***
APDU	Application Protocol Data Unit
CA	Conditional Access
CAM	Conditional Access Module
CI (+/plus)	Common Interface (plus)
DAE	Declarative Application Environment (see reference [OIPF_DAE2])
DASH	Dynamic Adaptive Streaming over HTTP
DRM	Digital Rights Management
DSLAM	Digital Subscriber Line Access Module
EPG	Electronic Programming Guide
FCC	Fast Channel Change
HbbTV	Hybrid Broadcast Broadband Television
LSC	Low Speed Communication (Channel)
NIT	Network Information Table
nPVR	Network-PVR
OSDT	Online Service Description Table
PVR	Personal Video Recorder
RET	Retransmission
SDT	Service Description Table
STB	Set Top Box
(MPEG-2) TS	(MPEG-2) Transport Stream

 Copyright 2014 © Open IPTV Forum e.V.

4 STB-less IPTV Architecture

Figure 1, shown below, gives a high-level block diagram for the CAM-based STB-less IPTV System.

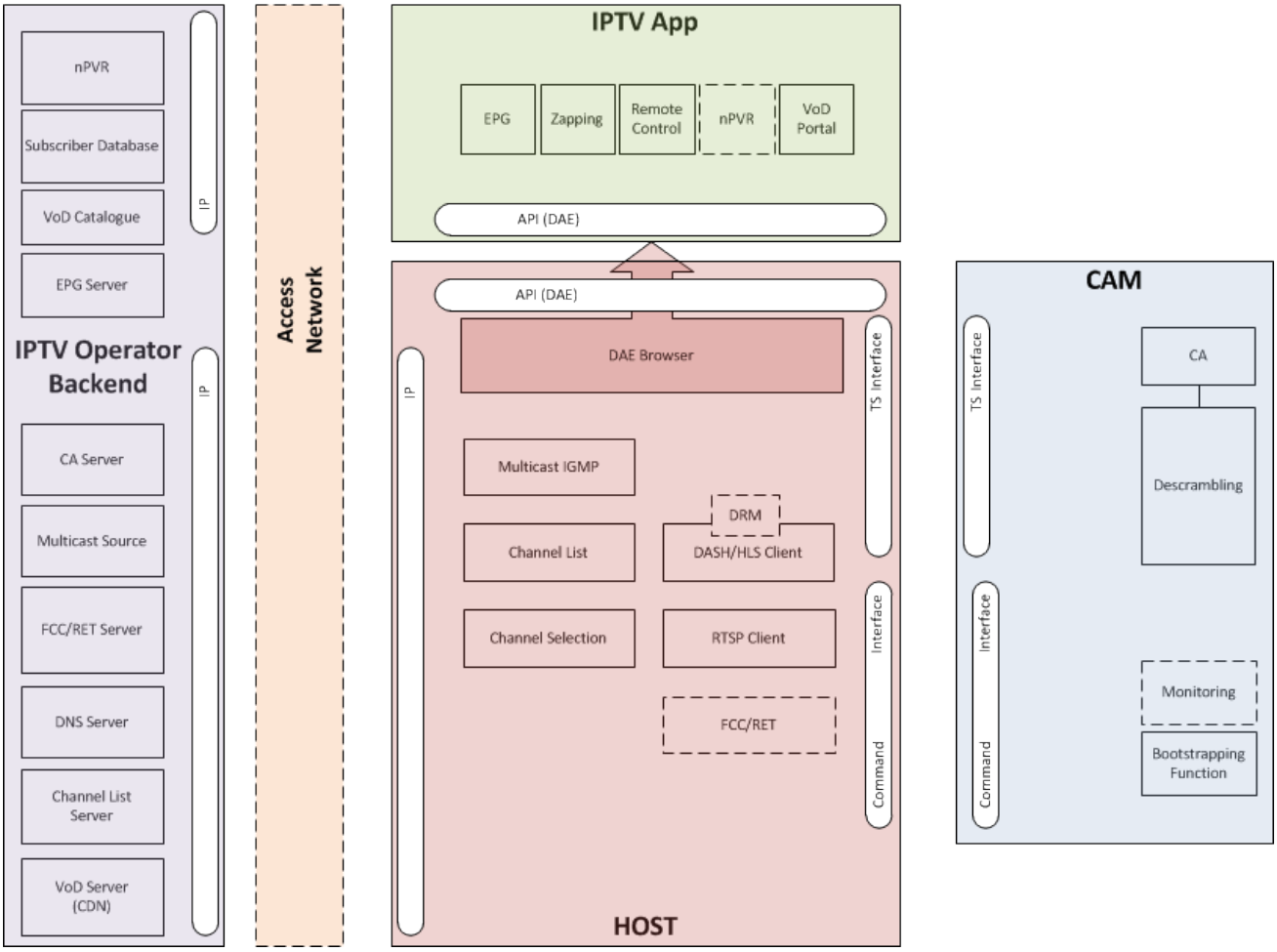

Figure 1: High-level block diagram including the major components making up the CAM-based STB-less IPTV System

The CAM-based STB-less IPTV architecture is divided into a number of high-level blocks (colored for sake of clarity):

- **IPTV Operator Backend**: Hosting all IPTV functions on the Operator side. Apart from making available the IPTV multicast streams, and providing the channel list, the backend includes such functionality as DNS servers for bootstrapping, FCC and RET servers and CA servers. The IPTV Operator Backend communicates directly with the Host via IP and (via the Host) with the CAM and the IPTV App running on the Host.

- **Host/TV**: The Host consists of that part of the TV which is provided by the TV manufacturer. Important functions are the Scheduled Content Services (Linear TV) client (based on the DAE video/broadcast object), the VoD client (based on the video/mpeg object) and a DAE browser running the IPTV App. Responsibilities of the Host in the STB-less IPTV System include acquiring multicast streams via IGMP, managing channel lists, decoding and playing back MPEG2_TS streams and optionally performing Fast Channel Change and RTP Retransmission. In addition, the Host communicates with the CAM module and provides the CAM with the possibility to access the internet by providing IP connectivity. The Host communicates with the IPTV Operator Backend via IP and with the CAM via the Command and TS Interfaces. Furthermore, the Host is capable of running DAE Applications (Apps) and communicating with these DAE Apps via (a subset of) the APIs described in [OIPF_DAE2] according to the profile described in section 5.

  In this document, the terms Host and TV can be used interchangeably

- **IPTV App**: The IPTV App runs on top of the Host platform, specifically in the DAE browser environment. The IPTV App is responsible for presenting the user interface the user sees when watching STB-less IPTV. This

 Copyright 2014 © Open IPTV Forum e.V.

includes presenting an EPG and allowing the user to switch channels using the remote control, and may include other features such as presenting a portal for interactive services. The IPTV App is built on the DAE platform provided by the TV and uses a subset of the existing OIPF DAE APIs (see the DAE profile in section 5). The IPTV App communicates with the Host via these APIs and with the IPTV Operator Backend via IP mediated by the Host. In addition, the IPTV App might communicate with the CAM via the OIPF DAE Content Service Protection APIs.

- **CAM (or CSPG-CI+)**: Within the STB-less IPTV System, the CAM has two main purposes: 1) Responsible for descrambling the MPEG2-TS streams received by the Host, and 2) Handling all operator-specific (non-standardized) communication with the IPTV Operator Backend that is outside the scope of the IPTV App (e.g. bootstrapping, obtaining channel list, etc). The CAM communicates with the Host via the TS and Command Interfaces. In addition, the CAM can access the internet via the so-called Low-Speed Communication (LSC) Resource. Furthermore, the CAM may communicate with the IPTV App via the OIPF DAE Content Service Protection APIs. In OIPF terms, the HNI-CSP interface defines the interface between the OITF and the CI+ CAM (the CSPG-CI+).

- **Access Network**: Included in the diagram for sake of completeness. At the moment, the Access Network is not assumed to contain any functionality specific for the STB-less IPTV System described in this document and is out-of-scope.

NOTE: Despite the fact that the various CI+ specifications support Hosts which have more than one CI+ interface, and thus support more than one simultaneous CAM, this document assumes any Host will include a maximum of one OSDT-supporting CAM.

4.1 CAM-less STB-less IPTV

In addition to the CAM-based system described in the previous section, an alternative option is to use the system without a CAM. Such a system is especially relevant in case unencrypted Linear TV Services are used. Alternatively, some CA or DRM system implemented on the Host could be used, however the selection and/or definition of such systems are outside the scope of this document.

In a CAM-less scenario, Service Discovery is handled by the Host, as defined in section.6.2.

 Copyright 2014 © Open IPTV Forum e.V.

5 DAE specification profile

In the STB-less IPTV system described in this document, the IPTV App is running inside an OIPF DAE browser environment as defined in [OIPF_DAE2].

Specifically, the Host SHALL support the subset of [OIPF_DAE2] as defined in Annex A of [HbbTV1.5].

In addition, the Host SHALL also support the following additional APIs, where the table below takes precedence in case of any conflicts with Annex A of [HbbTV1.5].

Table 1: DAE Specification Profile

Section, sub-section	**Reference in [OIPF_DAE2]**	**Mandatory (M) / Optional (O) / Not Included (NI)**	**Notes**
Video/Broadcast			
video/broadcast embedded object	7.13.1	M (see notes)	Access rules specified in section A2.4.2 of [HbbTV1.5] MUST NOT apply to the IPTV App.
Extensions for recording and timeshift	7.13.2	O (see notes)	If the 'localTimeshift' attribute in the capabilities XML provided by the Host indicates it supports localTimeshift, the Host SHALL support the constants, properties and methods specified in section 7.13.2 of [OIPF_DAE2], with the possible exception of the recordNow() and stopRecording() methods. The recordNow() and stopRecording() methods MAY be supported, but are not used by any features specified in this document.
Extensions to video/broadcast for current channel information	7.13.7	M (see notes)	The currentChannel property of the video/broadcast object SHALL be supported.
ChannelConfig object	7.13.9	M (see notes)	The following properties and APIs SHALL be supported: Properties: channelList, onChannelListUpdate(), currentChannel Methods: createChannelObject() In addition, the new createChannelListFromUri() defined in [STB-LESS-FP] SHALL be supported.
ChannelList class	7.13.10	M (see notes)	The getChannel method SHALL be supported, the other methods are not included.
Channel class	7.13.11	M	
Extensions to A/V Control object for DRM rights error	7.14.6	M	

 Copyright 2014 © Open IPTV Forum e.V.

Additional support for protected content	7.16.6	M	

5.1 User input

The Host SHALL provide a mechanism for the end user to generate key events according to section 10.2.2 of [HbbTV1.5]. The Host SHALL support the following Key Events as defined in Table 2 for IPTV Apps. For IPTV Apps this table takes precedence over the table in Section 10.2.2 of [HbbTV1.5].

Table 2: Key Events and their status

Button	**Key Event**	**Status**	**Different from HbbTV**
4 colour buttons (red, green, yellow, blue)	VK_RED, VK_GREEN, VK_YELLOW, VK_BLUE	Mandatory	No
4 arrow buttons (up, down, left, right)	VK_UP, VK_DOWN, VK_LEFT, VK_RIGHT	Mandatory	No
ENTER or OK button	VK_ENTER	Mandatory	No
BACK button	VK_BACK	Mandatory	No
Number keys	VK_0 to VK_9 inclusive	Mandatory	No
Play, stop, pause	VK_STOP and either VK_PLAY and VK_PAUSE or VK_PLAY_PAUSE	Mandatory	No
Fast forward, fast rewind	VK_FAST_FWD, VK_REWIND	Mandatory	No
Record	VK_RECORD	Optional	Yes
TEXT or TXT or comparable button	Not available to applications, handled by Host	n/a	No
Channel selection buttons (P+, P-)	VK_CHANNEL_UP, VK_CHANNEL_DOWN	Mandatory	Yes
Volume buttons	VK_VOLUME_UP, VK_VOLUME_DOWN	Optional	Yes
MENU button	VK_MENU	Optional	Yes
Guide button (EPG)	VK_GUIDE	Mandatory	Yes

If the IPTV App exits as a result of the mechanism in section 5.2.2 of [OIPF_DAE2] then the following shall apply:

- The IPTV App SHALL be re-started from the beginning.
- If the application was presenting broadcast video (via the video/broadcast object) at the time it exited then that video shall continue to be presented but shall be under the control of the Host as defined in clause H.2 of [OIPF_DAE2] (as if the release method was called on the video/broadcast object as the last operation before the application exited). This transition shall be seamless if the video was not scaled, cropped or positioned.
- If such a re-started IPTV App calls bindToCurrentChannel and video presented by its former instance is presented under the control of the Host then the video SHALL transition to being under the control of the application as defined in clause H.2 of [OIPF_DAE2] and the definition of the bindToCurrentChannel method.

 Copyright 2014 © Open IPTV Forum e.V.

6 Service Discovery & Provisioning

To start consuming Scheduled Content (Linear TV) services, the Host (and/or CAM) requires a channel list containing the location of each service and an EPG carrying metadata for each service. For the STB-less IPTV system described in this document, the channel list SHALL be defined by an XML-based OSDT file (Online SDT), standardized in CI+ 1.4 Annex D [CI+1.4].

Before retrieving the OSDT from the IPTV Operator Backend, first a bootstrapping (entry point discovery) process must be performed to acquire the location (in the form of a URI) from which the OSDT can be retrieved.

This specification foresees two methods for this bootstrapping process:

- CAM-based Service Discovery, where the bootstrapping is performed by the CAM
- CAM-less Service Discovery, where the bootstrapping is performed by the Host

NOTE: The OIPF Functional Architecture makes the distinction between Service Provider Discovery, in which a list of available IPTV Service Providers is obtained, and Service Discovery, in which a list of services available at a specific Service Provider are obtained. In the STB-less IPTV system described in this document, only the latter is described, i.e. how to obtain a list of services that are available at a specific IPTV Service Provider. That said, scenarios where multiple IPTV Operators are available to a specific Host can be supported at the CAM level.

6.1 CAM-based Service Discovery

6.1.1 Bootstrapping via DNS

Once the Host has been turned on, and the CAM has been initialized, the CAM SHALL start searching for IPTV services. The first step in this process is performing a DNS query to see if an OSDT server is available in the IPTV Operator Backend and find its IP address.

NOTE: The process described here assumes that the CAM has been provisioned by a specific IPTV Operator with the hostname of that operator's OSDT Server. Alternatively, the CAM could instead first contact a general bootstrapping server controlled by the operator. The CAM could then, using a custom protocol implemented on the CAM and thus outside the scope of this document, ask the bootstrapping server for the hostname of the IPTV Operator's OSDT server. The advantage of this more flexible approach is that the bootstrapping server could also be used for other purposes, such as upgrading the firmware in the CAM. Since such behavior requires custom and operator-specific functionality in the CAM, it is outside the scope of this document.

NOTE: The advantage of having the CAM be responsible for the DNS process instead of the Host is that it allows the DNS process to use extended DNS functionality, such as DNS SRV records, which may not be supported on the Host. With DNS SRV, it is no longer necessary for the CAM to be pre-provisioned for a particular operator, but it can instead search for a generic DNS service name, such as _oipf-stbless-iptv.

Before a CAM can send a DNS query, it first has to obtain the location of the DNS server(s) configured in the Host (which the Host has for example obtained through DHCP during network attachment). For this purpose, CI+ 1.4 has defined the new comms_IPconfig_request() and comms_IPConfig_reply() APDUs.

If the CAM intends to perform a DNS query, it first sends the comms_IPconfig_request() APDU over the Command Interface to the Host. Upon receiving the a comms_IPconfig_request() APDU, the Host SHALL reply with the comms_IPConfig_reply() APDU, containing, if it is connected, its network configuration details (IP address, network mask, gateway, and DNS server(s)).

After having obtained the location of the DNS server, the CAM MAY proceed with sending a DNS query for the hostname (in the form of a FQDN, e.g. osdt.iptv-operator.com) that it was either provisioned with or that it previously retrieved from a DNS SRV record. If this CAM sends a DNS query, it SHALL use the regular CI+ LSC Resource to do so. After receiving a reply from the DNS server, the CAM MAY store the obtained IP address for future use and SHOULD proceed with the Service Information procedure described below.

6.1.2 Constructing OSDT URL

After the CAM has obtained the IP address of the OSDT server as part of the bootstrapping procedure described in section 6.1.1, it SHALL proceed to construct the URL to the actual OSDT file using the retrieved IP address and

 Copyright 2014 © Open IPTV Forum e.V.

suffixing it with the path to the OSDT file which was pre-provisioned in the CAM (e.g. http://12.34.56.78/files/osdt.xml). In the STB-less IPTV system described in this document, it is assumed that the OSDT is retrieved via either HTTP or HTTPS.

NOTE: The process described here assumes that all that is needed to retrieve the OSDT file from the OSDT server is knowing the hostname of the server and the path to the file (e.g. /files/osdt.xml). An IPTV operator might want to protect access to the OSDT file by using various forms of authentication and access control (e.g. by extending the URL to the OSDT with a unique CAM identifier and checking this value upon reception of the request in the IPTV Operator Backend). Such functionality can be implemented as part of the Service Information procedure on the CAM; since it is the CAM which constructs the URL, it can include any information necessary as part of the query string part of the URL, e.g. “http://12.34.56.78/files/osdt.xml?subscriber_id=731613&cam_serial=ajhs10713a”. Since such behavior requires custom and operator specific functionality in the CAM, it is outside the scope of this document.

6.1.3 Sending OSDT location URI to Host via Operator Profile NIT

Once the CAM has constructed the URL that points to the OSDT on the IPTV Operator Backend (or on the CAM internal file system), it SHALL send the constructed URL to the Host via the method describe below, so that the Host is able to retrieve the OSDT.

In order to send the OSDT URL to the Host, the CAM SHALL use a CICAM NIT (Operator Profile NIT), specifically the new CI+ 1.4-defined Operator Profile Type 2 containing a uri_linkage_descriptor, see section 15.4 of [CI+1.4]. After having created the CICAM NIT, the CAM SHALL send the CICAM NIT to the Host using the operator_status() and operator_nit() APDUs over the Command Interface.

For details on how the CAM announces the availability of an Operator Profile or CICAM NIT, see CI+ 1.3 section 14.7 [CI+1.3].

6.1.4 Retrieval of OSDT by Host

The CI+ 1.4 specification allows for two methods of obtaining the OSDT file. The first is to have the CAM download or construct the OSDT file, store it on its local file system, send the Host a link to the file system on the CAM, and then have the Host retrieve the OSDT from there. The second method is to have the CAM construct the URL to the OSDT, send it to the Host, and then have the Host retrieve the OSDT itself. Unless stated otherwise, the rest of this document assumes the latter approach.

Once the Host receives the CICAM NIT from the CAM, it SHALL request the OSDT listed in the uri_linkage_descriptor of the NIT. If the uri_linkage_descriptor contains a url with a ‘http://’ or ‘https://’ scheme, it SHALL send an HTTP GET request for the particular URI (e.g. http://12.34.56.78/files/osdt.xml).

In some cases, the CAM MAY decide to provide the OSDT itself and make it available to the Host via the CAM’s file system. In such cases, the URI in the uri_linkage_descriptor in the NIT starts with ‘cicam://’. If this is the case, the Host SHALL use the operator_osdt_request() APDU to request the OSDT from the CICAM. Reasons for having the CAM provide the OSDT include the CAM creating it from scratch, the CAM making alterations to a generic OSDT (e.g. it can remove channels the user is not subscribed to from the OSDT), or the CAM downloading the OSDT in some proprietary fashion.

6.1.5 Exchanging capabilities between Host and CAM

The OSDT file lists a set of Services (i.e. channels), described by the Service element. Each Service can contain one or more ServiceLocation elements including a priority setting for each ServiceLocation (for the exact formatting of the OSDT file see Annex D of [CI+1.4]). A Service Location refers to a specific method through which a particular Service can be obtained, described in the form of an xml structure. For example, in case a particular linear TV channel can be obtained through both IP Multicast and unicast MPEG DASH, that channel is listed as having two Service Locations.

When the Host has obtained the list of channels and associated Service Locations, it SHALL verify, based on the Service Location Type, which of these Service Locations it is able to receive (e.g. the Host might not understand RTSP and thus will not be able to receive any Service Locations with the RTSP ServiceLocationType). It SHALL then compile a list of all the Service Locations it does not know how to receive and asks the CAM over the Command Interface whether it is able to handle those Service Locations. For this purpose, CI+ 1.4 defines the new player_verify_req() APDU. The Host SHALL sequentially send this APDU for each Service Location it does not know how to handle.

Upon receiving the player_verify_req() APDU, the CAM SHALL check, based on the Service Location Type, whether it is able to handle the indicated Service Location Type. It might be that the CAM does understand RTSP, while the Host

 Copyright 2014 © Open IPTV Forum e.V.

does not. After checking a particular Service Location, the CAM SHALL, according to [CI+1.4], return a player_verify_reply() APDU including a boolean that tells the Host whether the CAM can handle the particular Service Location.

After looping through all Service Locations, the Host will have a full list of Service Locations it is able to offer to the user, with some of them that can only be handled by the CAM, and the remainer which the Host can handle itself. It might be that a particular Service (i.e. channel) is available using two Service Locations (e.g. via IP Multicast and via unicast MPEG DASH). In these cases, the Host SHALL include the Service Location with the highest priority (as indicated by the OSDT) in any channel lists which may be created based on the OSDT. The Host SHALL NOT include any Service Locations with a lower priority if a Service Location with a higher priority is also available and is verified to be able to be handled by either the CAM or the Host.

Once the Host has derived a full list of supported Services it can use these to construct or modify an existing channel list and offer the updated channel list to the IPTV App (see section 8).

6.1.6 Service Discovery Information Updates

At some point in time, an update to the OSDT might be available at the IPTV Operator Backend (e.g. with added/removed channels). It should be noted that an IPTV Operator might decide to either update the OSDT at the existing OSDT URL, or update the OSDT URL as well. There are three distinct mechanisms which may be used to update the channel list in the Host:

- Initiated by the Host. Periodic polling by the Host for a new version of the OSDT, as described in section 15.4.1 of [CI+1.4], might lead to the Host discovering that an updated OSDT is available. The polling interval is defined by the `min_polling_interval` parameter, see [CI+1.4].

- Initiated by the CAM. A CAM MAY use any custom and/or proprietary mechanism to keep in touch with the IPTV Operator Backend (e.g. periodic polling, bidirectional communication channel, etc.). As part of this mechanism, which is out of scope of this document, the CAM may be notified by the IPTV Operator that a new version of the OSDT is available. After the CAM has confirmed that a new OSDT is indeed available, it MAY announce this to the Host via Operator Profile Type 2 in a manner similar to how it announced the initial OSDT (see section 6.1.3).

- Initiated by the IPTV App. The IPTV App MAY use any custom and/or proprietary mechanism to keep in touch with the IPTV Operator Backend (e.g. periodic polling, bidirectional communication channel, etc.). As part of this mechanism, which is out of scope of this document, the IPTV App may be notified by the IPTV Operator that a new version of the OSDT is available. After the IPTV App has confirmed that a new OSDT is indeed available, it SHALL announce this to the Host via the newly-defined createChannelListFromUri() API (see section 7.13.9.2 of [OIPF_DAE2]). As an attribute, the IPTV App will include the URL to the OSDT.

Upon receiving notice of the new OSDT, either directly, via a CICAM NIT or via the createChannelListFromUri () API, the Host SHALL respond by retrieving and parsing the OSDT as described in sections 6.1.4 and 6.1.5. In the case where any new Services and ServiceLocations have been added to the OSDT, it might be necessary for the Host to check with the CAM whether it is able to handle some of these new Services (see section 6.1.6). If new Services or Service Locations are found, and such Service Locations are confirmed to be able to be handled by either the CAM or the Host, the Host SHALL make these new Services available to the IPTV App if and when it sends the next getChannelConfig() request.

NOTE: In some cases, a new OSDT file might include a reference to an updated IPTV App, either at a different URL than the earlier version (which makes the change immediately apparent to the Host), or at the same URL (which doesn't make the update immediately apparent to the Host and therefore requiring e.g. a HTTP HEAD request). In both scenarios, if a new version of the IPTV App is confirmed to be available it is RECOMMENDED that the Host keep showing the existing IPTV App until the next time the Host is rebooted or the IPTV App is restarted in some other fashion. This prevents an unfriendly user experience.

NOTE: Theoretically, it is possible for the new OSDT to have removed any reference to the Service which is currently being watched by the user (e.g. the particular channel might be removed from the IPTV Operator's channel line-up). It is up to the implementation to decide how to deal with this scenario, e.g. to automatically switch to a different channel or to show an error message.

 Copyright 2014 © Open IPTV Forum e.V.

6.1.7 Removal of CAM

If, at any point in time, the user decides to remove the CAM from the CI+ slot, the IPTV service is assumed to have been discontinued from a Host's perspective. In this case, the Host MUST stop the IPTV App if it is still running, SHALL remove the App and any associated data from its memory and MUST remove that part of its internal Channel List which is associated with the OSDT provided by the CAM. In practice this means that any Channels of the type ID_IPTV_OSDT (see section 8.2.1), will have to be removed from the Host's internal Channel List (assuming one OSDT).

NOTE: In some hybrid cases, a particular Channel might be offered both via IPTV as well as a traditional DVB broadcast. Upon parsing the OSDT, the Host may for some reason decide that this is the case, and remove the DVB Channel from the channel list in favor of the IPTV version. If this is the case, and the CAM is removed, the Host is expected to reinstate the DVB Channel to the channel list. Such host-specific mechanisms are considered to be out-of-scope of this document.

6.2 CAM-less Service Discovery

6.2.1 Bootstrapping via DNS

In a CAM-less scenario, see section 4.1, the Service Discovery process is carried out by the Host instead of the CAM. The main difference between a Host-centric scenario and a CAM-centric scenario is that in the CAM-centric case, there exists the possibility to provision the CAM for a specific IPTV Operator, allowing the Service Discovery process to start from some configured parameter to that IPTV Operator, such as a hostname or IP address. A Host device, however, should work across a wide variety of IPTV Operators, the full list of which is likely to change over time and might not be known during time of manufacture. As such, a Host device can not be expected to be pre-provisioned with the entry points for IPTV Operators available to the user.

Instead of using the pre-provisioned hostname of the IPTV Operator's entry point as a basis for the Service Discovery process, in a CAM-less scenario, the Host SHALL use the DNS SRV mechanism defined in section 16 of [OIPF_PROT2], see also [STB-LESS-FP]. Once the Host device is triggered to start the Service Discovery process, it SHALL perform a DNS query for the service name *_oipf-osdt-iptv*. The IP address returned in the DNS SRV record is associated with the IPTV Operator's OSDT server.

In order for the CAM-less Service Discovery process described in this section to function correctly, it is necessary that the Host is configured with a DNS server that contains a DNS SRV record that associates the service name _oipf-osdt-iptv._tcp.dvb.org with the OSDT Server of the local IPTV Operator. In a typical scenario, the Host will receive the location of the DNS server via DHCP from the local network's Residential Gateway (which in turn will have received it from the local network operator). If, for some reason, the user has configured the Host (or the Residential Gateway) to use a different DNS server, e.g. a global one, the DNS SRV record will not result in the correct IPTV Operator entry point, since the global DNS will not refer the Host's DNS SRV request back to the user's local IPTV Operator. In the case where the Host determines this to be the case (e.g. because it is unable to download a correct OSDT file), the Host MAY allow the user to input the IP address of the OSDT server manually.

6.2.2 Constructing OSDT URL

After the Host has obtained the IP address of the OSDT server as part of the bootstrapping procedure described in section 6.2.1, it SHALL proceed to construct the URL to the actual OSDT file using the retrieved IP address appended by '/osdt.xml' (e.g. http://12.34.56.78/osdt.xml).

6.2.3 Retrieval of OSDT by Host

The Host SHALL request the OSDT by sending an HTTP GET request for the OSDT URL.

6.2.4 Service Discovery Information Updates

In the CAM-less discovery process, the following mechanism MAY be used by the IPTV App to update the channel list in the Host:

- Initiated by the IPTV App. The IPTV App MAY use any custom and/or proprietary mechanism to keep in touch with the IPTV Operator Backend (e.g. periodic polling, bidirectional communication channel, etc.). As part of this mechanism, which is out of scope of this document, the IPTV App may be notified by the IPTV Operator that a new version of the OSDT is available. After the IPTV App has confirmed a new OSDT is indeed

 Copyright 2014 © Open IPTV Forum e.V.

available, it SHALL announce this to the Host via the newly-extended createChannelListFromUri () API (see section 7.13.9.2 of [OIPF_DAE2]). As an attribute, the IPTV App will include the URL to the OSDT.

Upon receiving notice of the new OSDT, via the createChannelListFromUri () API, the Host SHALL respond by retrieving and parsing the OSDT. If new Services or Service Locations are found, and such Service Locations are confirmed to be able to be handled by the Host, the Host SHALL make these new Services available to the IPTV App if and when it sends the next getChannelConfig() request.

NOTE: In some cases, a new OSDT file might include a reference to an updated IPTV App, either at a different URL than the earlier version (which makes the change immediately apparent to the Host), or at the same URL (which doesn't make the update immediately apparent to the Host and therefore requires e.g. a HTTP HEAD request). In both scenarios, if a new version of the IPTV App is confirmed to be available it is RECOMMENDED that the Host keep showing the existing IPTV App until the next time the Host is rebooted or the IPTV App is restarted in some other fashion. This prevents an unfriendly user experience.

NOTE: Theoretically, it is possible for the new OSDT to have removed any reference to the Service which is currently being watched by the user (e.g. the particular channel might be removed from the IPTV Operator's channel line-up). It is up to the implementation to decide how to deal with this scenario, e.g. to automatically switch to a different channel or to show an error message.

6.3 Bootstrapping the IPTV App

Apart from including the full list of Scheduled Content Services offered by the IPTV Operator, the OSDT may also include information regarding an available IPTV App (Service Provider related application), which the Host MAY start upon parsing the OSDT, as specified in section 5.2.4 of [OIPF_DAE2]. For this purpose, an IPServiceList element in the OSDT can contain an IPService element without associated ServiceLocation elements but with an ApplicationLocation element, indicating the location from which the IPTV App may be retrieved. This element SHALL be unique in the IPServiceList element. An application that is declared as associated with a service in the OSDT follows the same rules as an application announced as associated to a scheduled content service (i.e. a broadcast related application) whereas section 5.2.3 of [OIPF_DAE2] is concerned.

It is up to the Host and thus beyond the scope of this document to specify how the IPTV App is integrated in the user interface of the Host and is made accessible to the user (e.g. as an input 'source' similar to HDMI or a native DVB tuner).

 Copyright 2014 © Open IPTV Forum e.V.

7 Service Presentation

7.1 EPG

The EPG is retrieved, rendered and managed by the IPTV App. Since the EPG is completely within the domain of the IPTV App, the EPG format can be operator-specific and proprietary.

However, in order for the IPTV App to know to which Service it should tune (or request the Host to tune to) when the user selects a program from the EPG, it is RECOMMENDED that there is some relationship between the EPG and the channel list provided by the OSDT. For these purposes, the OSDT format defined by CI+ 1.4 includes the DVBTriplet element that can be used to refer to a service event if the service is not delivered in a TS. Whatever operator-specific EPG format is therefore chosen, it MAY include an element that links a particular channel to a DVB Triplet-format identifier. Whenever the user selects a program or channel from the EPG, the IPTV App can look up the DVB Triplet for that entry in the EPG and check the OSDT whether a matching DVB Triplet can be found, thereby finding the associated Service and ServiceLocation(s).

NOTE: In the STB-less IPTV system described in this document, Service listed in the OSDT are not necessarily delivered using DVB-compatible MPEG-2 Transport Streams. As an example, Services might instead be delivered via ISOBMFF-based MPEG DASH. As such, an actual DVB Triplet can therefore not necessarily be defined. However, any unique combination of three values or strings, formed as a DVB triplet and listed in both the EPG and OSDT, can still perform the function of a unique channel identifier for the purpose of linking the OSDT and EPG. In the case of ISOBMFF-based MPEG DASH, for example, the IPTV Operator can decide on three numbers or strings which it will use as the DVB Triplet.

 Copyright 2014 © Open IPTV Forum e.V.

8 Service Selection & Play

Service Selection & Play refers to the process of actually requesting and receiving a particular Service, decrypting it, and playing it out on the TV.

The mechanisms described in this section mostly rely on the `video/broadcast` object and its related APIs defined by OIPF DAE [OIPF_DAE2].

8.1 Media delivery mechanisms

For the purposes of Scheduled Content Services, the Host device SHALL support at least the following media delivery mechanism via the video/broadcast object:

- IP Multicast of MPEG2-TS streams sent over RTP/UDP, without SIP session management. Defined in section 8.1.1 of [OIPF_PROT2]
- MPEG DASH according to section 4.3 of [OIPF_HAS2]

For streams delivered via IP Multicast or MPEG DASH, the Host SHALL support routing any encrypted streams over the CI+ 1.4 interface for decryption in the CAM.

In addition, the Host MAY support:

- RTSP according to the RTSP Profile defined in section 7.1.1 of [OIPF_PROT2]

NOTE: While this document does not require support for RTSP for Scheduled Content Service in combination with the video/broadcast object, RTSP is mandatory for interactive service handled by the AV Control Object, as defined in section 9.

8.2 Selecting services

The first step in the Service Selection & Play process is the user selecting a particular Service, be it a ‘live’ linear TV Service or a VoD Service. There are various methods with which a user can select a particular Service, such as the user pressing the ‘channel up/down’ buttons on the remote, using one of the numbered remote control buttons, or the user selecting a particular Service through the EPG.

NOTE: The rest of this section assumes services being advertised through the channel list generated by the OSDT. In addition, it is possible for services to be advertised through different means, such as a proprietary-formatted VoD catalogue retrieved and shown by the IPTV App. For such use cases, see section 9 on Interactive Services.

Irrespective of which method the user uses to select a service, this user interaction needs to be correlated back to a particular Service as listed in the OSDT.

The rest of this section assumes that the IPTV App is in charge of the user experience, and thus of the zapping experience. This means that the IPTV App will receive user input through the remote control and send the Host the commands to switch to a particular channel. Note that the Host therefore does not have to contain a native service selection mechanism. In addition, it assumes the Host will not kill the IPTV App following a channel change initiated by the IPTV App itself.

8.2.1 Communicating Channel List between Host and IPTV App

Before the IPTV App can send the Host a command to switch to a particular Service, the IPTV App first needs to be aware of which Services are available (i.e. the channel list). The App can request this channel list from the Host by sending the getChannelConfig() request, specified in Section 4.8.1.1 of the OIPF DAE [OIPF_DAE2], to the Host. The Host SHALL construct the requested channel list based on the retrieved OSDT (according to the guidelines for creating a Channel List as specified in [OIPF_DAE2]) and send it to the IPTV App upon receiving a getChannelConfig() request.

NOTE: In OIPF DAE a Channel contains a Channel Type field that is set to the type of Channel (e.g. DVB-S, DVB-C or ID_IPTV_SDS). For Channels created using an OSDT file, this field is set to the newly-defined ID_IPTV_OSDT (see [STB-LESS-FP]), irrespective of the ServiceLocationType listed in the actual OSDT file. Since the OSDT defined in [CI+1.4] is based on the DVB SD&S Record, the information that is conveyed as part of the ID_IPTV_OSDT Channel

 Copyright 2014 © Open IPTV Forum e.V.

Type is similar to that conveyed by ID_IPTV_SDS. For an overview of how to construct a Channel object from an OSDT Service Location, see section 4.5 in [STB-LESS-FP].

8.2.2 Communicating a selected Linear TV Service from the IPTV App to the Host

At some point after the IPTV App has received the Channel List from the Host and presented it to the user, a user will select a particular channel. At this point, the App needs to send a request to the Host to switch to that particular channel.

When a Channel switch is required, the IPTV App SHALL send a setChannel() request, specified in section 7.13 of [OIPF_DAE2], to the Host. As an attribute to the setChannel() request, the App SHALL include an identifier of the Channel object (as received as part of the Channel List exchange described in the previous section) representing the selected channel.

In response to the setChannel() request, the Host SHALL tune to the requested channel. Note that in order to properly switch to the requested channel, it might be necessary for the Host to correlate the requested Channel back to a Service in the OSDT (e.g. for retrieving additional information necessary for switching to that channel such as FCC/RET server addresses).

8.3 Accessing linear TV services

For the purposes of the STB-less IPTV System, four categories of 'Scheduled Content Services' (Linear TV) (as opposed to VoD) Services can be distinguished:

1. Scheduled Content Services received via IP Multicast and handled by the Host
2. Scheduled Content Services received via MPEG DASH and handled by the Host
3. Scheduled Content Services received over other delivery protocols, either standardized or proprietary, which are handled by the Host.
4. Scheduled Content Services received over other delivery protocols which might be unknown to the Host and handled by the CAM

A Host MUST support Scheduled Content Services received via IP Multicast (option 1) and MPEG DASH (option 2), according to section 8.1. In addition, a Host MUST support CICAM Player Mode (see [CI+1.4]) to allow for the CAM to handle other delivery mechanisms (option 4). A Host MAY support alternative delivery mechanisms itself (option 3).

Sections 8.3.1 and 8.3.2 describe the first types of Services (IP Multicast and MPEG DASH). Sections 8.3.3 and 8.3.4 deal with other Service delivery mechanisms , by the Host and the CAM respectively.

Upon receiving a setChannel() request from the IPTV App, the Host SHALL determine whether the Service indicated by the Channel object will handled by the Host or the CAM. It SHALL then start the process of Service selection with the appropriate mechanism.

8.3.1 Linear TV via IP Multicast handled by the Host

8.3.1.1 Joining channel

If the Channel object referenced in the setChannel() request is associated with an IP Multicast Service Location, the Host SHALL join the IP Multicast Service, Depending on the format of the Service Location listed in the OSDT, the Host SHALL send either an IGMPv2 or IGMPv3 Join request to the IP Multicast address listed as part of the Service location.

8.3.1.2 Format of Scheduled Content Services sent over IP Multicast

The Linear TV IP Multicast channels being used for the STB-less IPTV system described in this document SHALL consist of MPEG2-TS streams sent over RTP/UDP, see section 8.1.1 of [OIPF_PROT2]

8.3.1.3 Descrambling via CAM

In case of a scrambled service, after receiving the IP Multicast stream, the Host SHALL strip off the RTP headers, concatenate the resulting MPEG2-TS packets and pass the stream to the CAM for descrambling.

 Copyright 2014 © Open IPTV Forum e.V.

After stripping of the RTP header, the resulting MPEG2-TS stream is similar to MPEG2-TS streams received using other delivery mechanisms such as DVB-C (with the main difference being that there are no SDT and NIT packets present in the stream). In particular, the structure of the MPEG2-TS allows for the CAM to prepare for decryption ahead of time and it does not require buffering in the CICAM. Typically, the MPEG2-TS contains conventional ECM sections. For this reason, after the RTP headers have been stripped, the mechanisms for communicating with the CAM, descrambling and playing out the stream as specified by CI+ 1.3 [CI+1.3] apply.

8.3.2 Linear TV via MPEG DASH handled by Host

If the Channel object referenced in the setChannel() request is associated with an MPEG DASH Service Location Type, the Host SHALL attempt to access the MPEG DASH MPD. If the Host determines the format of the MPEG DASH stream to be according to section 4.3 of [OIPF_HAS2], the Host SHALL playout the MPEG DASH stream. If the Host determines the format of the MPEG DASH stream to be different from what is specified in section 4.3 of [OIPF_HAS2], the Host MAY playout the MPEG DASH stream.

8.3.2.1 Descrambling via CAM

If the received MPEG DASH stream is of a format compatible with section 4.3 of [OIPF_HAS2], the Host SHALL be able to send the MPEG DASH stream to the CAM for decryption according to the mechanisms specified in section 7 of [CI+1.4].

8.3.3 Linear TV via other delivery mechanisms handled by the Host

In addition to supporting IP Multicast, a Host might optionally support more delivery mechanisms, either standardized or proprietary, suitable for the delivery of Linear TV. Examples of such mechanisms are IP Multicast extended with Fast-Channel Change and/or Retransmission technology, a live profile of MPEG DASH, or RTSP.

Since the nature of any additional delivery mechanisms can vary, the details of how to access Services provided via these mechanisms is out of scope of this document.

8.3.4 Linear TV handled by CAM

As discussed in section 6.1.5, some Services might not be handled by the Host but by the CAM. Examples of such Services are those using FCC/RET which might not be implemented in the Host, services being streamed using proprietary streaming mechanisms, or future standard distribution methods not supported by the TV. In this case, the Host SHALL request the CAM to switch to a particular Service after receiving a setChannel() request from the IPTV App. For these purposes, CI+ 1.4 [CI+1.4] includes a set of Player Resource APDUs that allow a Host to start and control content playout on the CAM.

In order to request the CAM to play a particular Service, the Host SHALL send a player_play_req() APDU to the CAM. Included in the APDU is a link (in the form of a byte range) to a particular Service Location in the OSDT. The CAM, which has earlier indicated that it is able to handle the particular Service Location during the process described in section 6.1.5, SHALL proceed by setting up the necessary player instance and replying with a player_session_start_req(). For more details on the mechanisms that the Host can use to further control playback by the CAM, such as pausing or stopping the session, see section 8 of [CI+1.4].

NOTE: Once a CAM has received the player_play_req() message, it starts requesting the streams indicated by the particular Service Location. The exact method used for this purpose is protocol specific and therefore not discussed in this document in detail. The most important aspect of this is the concept of Hybrid IP Sessions as introduced in CI+ 1.4. A Hybrid IP Session allows the CAM to use the Command Interface (via LSC) to request IP content, while receiving the requested content over the TS Interface. The advantage of this system is that it allows the CAM to retrieve content that does not fit over the Command Interface due to that interface's bandwidth limitations.

Alternatively, when the CAM has some independent connectivity function (e.g. an external WiFi interface), the requested content can be obtained directly by the CAM without relying on the LSC Hybrid facility.

8.4 Re-launch of IPTV App

There are various reasons for the Host shutting down the IPTV App, most notably a TV power cycle, a user switching to another source on the TV or the user using some other TV feature not related to the IPTV Service. This document does not specify the exact trigger mechanisms which might cause the Host to subsequently launch the IPTV App again at a later instant. Depending on the implementation, this might be the case when the Host is turned on again after a power

 Copyright 2014 © Open IPTV Forum e.V.

cycle with the IPTV App being active when the Host was turned off or the user switching to a input source associated with the IPTV App. It is up to the implementation to decide on the details of these triggers and the position of the IPTV App in the Host GUI.

Irrespective of the reason it was (re)launched, upon starting the IPTV App it might be desirable from a user perspective to find it in the state he last left it in. A prime example of this is the IPTV App automatically switching to the last-viewed channel, but it might also include language settings, VoD favorites lists, etc. In the STB-less IPTV system described in this document, such behavior is at the discretion of the IPTV App to implement or not. Since the IPTV App is in full control of channel tuning, it can simply send a setChannel() request to the Host upon launch.

The only element which is needed on the Host side to support such functionality is the ability for the IPTV App to store some simple key-value information in non-volatile memory. Since the IPTV App is running in a CE-HTML browser environment, the simplest way to achieve such storage on the part of the IPTV App is using cookies. Using a JavaScript setCookie() method, the IPTV App is able to store any information it might need on the Host.On the Host side this requires that the browser support cookies for at least the IPTV App.

NOTE: If the Host supports the Web Storage APIs defined in section 7 of [OIPF_DAE2-WEB], these APIs might present an alternative solution for storing data on the Host. However, this document DOES NOT mandate the use of the Web Storage APIs and it is up to the IPTV App to check whether a given Host supports these APIs.

8.5 Radio services

In addition to providing linear TV services, the STB-less IPTV system described in this document is also suitable for the delivery of radio (i.e. audio-only) services. Such services can be signaled in the OSDT via the ServiceType field in the IPServiceType element. The Host SHALL parse such services in the OSDT and SHALL make them available via the channel list provided to the IPTV App upon receiving a getChannelConfig() request.

8.6 Teletext services

If the Host supports Teletext according to section 7 of [OIPF_MEDIA2] then it shall include a mechanism to start the Teletext envrionment. This mechanism may co-exist with the capability to start and stop a broadcast-related DAE application instead of analogue teletext services defined in section 5.2.3.1 of [OIPF_DAE2]. For example, if both analogue teletext and a teletext replacement DAE application are present in a service or channel then pressing a "text" button MAY cycle through 3 states - TV -> DAE application replacing teletext -> analogue teletext -> TV -> DAE application and so on.

While in the Teletext environment, the Host SHALL be in control of all buttons associated with Teletext. Upon the user exiting the Teletext environment, the Host SHALL return control of the buttons to either the IPTV App (according to section 5.1) or to a broadcast-related DAE application depending on what is running at that point.

 Copyright 2014 © Open IPTV Forum e.V.

9 Interactive Services

In the scope of this document, Interactive Services refers to the selection and playout of any media stream that does not fall under the Scheduled Content Services banner. Examples of services that fall under Interactive Services include VoD and network-based PVR (nPVR).

Since network-based services like nPVR are tightly integrated with a specific operators backend, the service presentation and selection of Interactive Services is handled completely by the IPTV App, and can thus be handled in a proprietary fashion. After the IPTV App is bootstrapped and started as described in section 5, the IPTV Operator is free to include in the IPTV App a portal that leads to a variety of Interactive Services.

The scope of this section is to describe how the playout of these Interactive Services is handled after the user has made his selection. Since the playout of these services is handled by the Host, possibly with the CAM handling any DRM schemes, these aspects need to be handled in a standardized manner.

This section will discuss how the IPTV App can initiate playout of services in the Host which are not signaled in an OSDT file, with a focus on MPEG DASH and RTSP-based Interactive Services.

9.1 Initiating playout of non-OSDT services

Section 8 describes how the IPTV App can ask the Host to switch to a particular Scheduled Content Service (i.e. channel). It does so using the OIPF DAE video/broadcast object and the setChannel() command including a reference to an IPService element listed in an OSDT file. In the case of Interactive Services however, which are not listed in an OSDT file but are described in a proprietary format exchanged between the IPTV Operator Backend and the IPTV App, there is no OSDT file to which the IPTV App and the Host can refer.

The OIPF DAE [OIPF_DAE2] describes how Linear TV playout is handled via the video/broadcast object and set of APIs. In the same way, the AV Control (or video/mpeg) object (see [OIPF_DAE2]), can be used to control playback of non-live streaming media, such as those used for Interactive Services.

Unless specified otherwise in the following sections, all playback of Interactive Services described in this document is performed using the AV Control object, including the extensions specified for it in the OIPF DAE.

9.1.1 Supported delivery mechanisms

For the purposes of Interactive Services, the Host device SHALL support at least the following media delivery mechanism via the AV Control object:

- RTSP according to the RTSP Profile defined in section 7.1.1 of [OIPF_PROT2]
- MPEG DASH according to section 4.3 of [OIPF_HAS2]

9.1.2 RTSP

In the case where an Interactive Service is signaled via RTSP and handled by the Host, the current specifications for controlling an RTSP stream via the AV Control object, as specified by [CEA] and [OIPF_DAE2] apply.

In some cases, however, the Host might not be able to handle the type or variant of RTSP offered. In such scenarios, the CAM might be able to handle the RTSP stream instead, similar to the situation described in section 8.3.2.

To trigger the CAM to start playback of an RTSP-controlled stream, the Host uses the same set of APDUs as used to trigger the CAM to start playback of a Scheduled Content Service (e.g. player_play_req()), see section 8.3.2. However, in contrast with the Scheduled Content Services case, with Interactive Services there is no OSDT file containing the ServiceLocation element(s) describing the particular media stream. Since the APDUs to control playback on the CAM are based on this concept of a ServiceLocation (see section 6.1.5), it is necessary for the Host to create a new ServiceLocation object based on the information it received from the IPTV App. To do so, it uses the information passed to the AV Control object (i.e. the RTSP URI), basically translating between the two interfaces.

Before using the player_play_req() APDU to ask the CAM to playback the RTSP-controlled stream, it is recommended that the Host first verifies that the CAM can actually handle RTSP. This verification procedure is handled in the same manner as described in section 6.1.5 for Linear TV Services (e.g. by passing a ServiceLocation object to the CAM and asking whether it is able to handle that ServiceLocation).

 Copyright 2014 © Open IPTV Forum e.V.

9.1.3 MPEG DASH

In the case where Interactive Services are delivered via MPEG DASH, the specifications relating to MPEG DASH in [OIPF_PROT2] and [OIPF_HAS2] apply.

In the case where a DRM scheme is used, the IPTV App can optionally include a contentAccessDescriptorURL, representing a link to a DRM scheme (e.g. a Marlin action token) , in the setChannel() request to the Host.

9.2 Specific Services

This section describes a number of value-added services which are commonly provided as part of an IPTV offering. Most of these services can be handled on the IPTV App level, which means they can make use of proprietary interfaces between the IPTV App and the IPTV Operator Backend.

9.2.1 Network-based PVR

NOTE: This section does not specify any new requirements for either the Host or the CAM, and as such, it can be considered informative.

Network-based PVR (nPVR) functionality can be divided into two distinct sets of features.

The first category relates to the everything that has to do with the scheduling of recordings - both from a GUI perspective as well as the communication with the IPTV Operator (nPVR) Backend - to the management of earlier recordings by browsing through them and selecting and/or deleting them. Since this functionality can be handled in a browser environment, and no communication with the TV and CAM is necessary, it is completely within the domain of the IPTV App. In order to communicate with the IPTV Operator Backend, the IPTV App can use standard web communication methods, such as HTTP, in combination with any proprietary protocols and data models as long as it is possible to implement those in a browser environment.

The second category relates to the playback of previously-recorded streams on the Host. For this purpose, the generic functionality for playout of non-OSDT services based on the AV Control object, described in section 9.1, can be used. Depending on the streaming technology used by the IPTV Operator, e.g. RTSP, MPEG-DASH or something else, the resulting content stream is handled by either the Host itself or the CAM. The IPTV App can control playback of the content via the standardized AV Control object APIs.

Since both types of functionality can be provided using the architecture and methods described in the rest of this document, no specific functionality needs to be added in order to support nPVR.

9.2.2 Network-based timeshift

NOTE: This section does not specify any new requirements for either the Host or the CAM, and as such, it can be considered informative.

As is the case with nPVR, described in the previous section, network-based timeshift as described by this section is orchestrated by the IPTV App. Compared to nPVR, however, network-based timeshift is slightly more complex due to the context switch from multicast to unicast without first going through a portal or GUI.

NOTE: It should be noted that since the goal of this document is to create a lightweight STB-less IPTV system, which leaves the highest possible number of features up to the IPTV App, this document does not mandate the use of the OIPF Timeshift functionality described in [OIPF_DAE2]. Instead, a simpler mechanism based on a combination of the video/broadcast and the AV Control object is proposed.

NOTE: Implementing network-based timeshift in the IPTV App can be quite complex from a user-experience perspective. There are a lot of buttons involved and the time it takes to do the context switch between the multicast stream (and the associated video/broadcast object) and the unicast stream (via the AV Control object) can have a large effect on the perceived user experience. This section does not pretend to present a definitive implementation guideline on how to implement network-based timeshift within the IPTV App. It merely gives an example on how network-based timeshift can be accomplished within the STB-less IPTV system described in this document without specificying any new functionality on the Host side.

The general notion behind the network-based timeshift mechanism described here is that from the moment the user presses one of the ‘trickplay’-buttons on his remote control associated with timeshift functionality (i.e. Pause, Fast-Forward, Rewind, etc.), the IPTV App switches from a video/broadcast player used for controlling linear multicast

 Copyright 2014 © Open IPTV Forum e.V.

content, to an AV Control object used for controlling playback of on-demand unicast content. From that point onward, the focus of the IPTV App remains in the AV Control object until the user decides to go back to 'live' content, for example by pressing the 'Stop' button. At that point, the video/broadcast object is re-started based on the multicast stream.

The mechanism used to stream the timeshifted unicast version of the media stream from the IPTV Operator Backend to the Host can be proprietary and/or operator-specific. As an example, the RTSP or MPEG DASH methods described in section 9.1 could be used.

Once the user presses one of the 'trickplay' buttons that indicate the use of timeshift functionality, the IPTV App uses some proprietary and/or operator-specific mechanism to retrieve the location of a unicast version of the channel currently being watched by the user. Alternatively, such a unicast version of a TV Service might already have been provisioned in the IPTV App via either an OSDT Service Location or some proprietary mechanism. Depending on the network-based timeshift mechanism employed by the IPTV Operator, such a unicast stream might either by a generic one, accessed by multiple users simultaneously and within which the user is free to navigate, or a 'dedicated' stream created for that specific user, where the IPTV Operator is in control of any trickplay functionality which may or may not be allowed for that stream.

After the location of the unicast stream is available, the IPTV App creates an AV Control object and initializes it with the location of the unicast stream (see section 9.1). If necessary, the IPTV App may use information received from the IPTV Operator Backend to seek to the correct position within the unicast stream. After the AV Control object is initialized and pushed to the foreground, the user is able to use his remote control to perform any (allowed) trickplay operations on the content, resuming the video as he sees fit.

If, at any point while consuming the unicast stream, the user decides to go back to the 'live' version of the TV Service, the IPTV App stops the AV Control object and resumes the video/broadcast object (if it was pushed to the background) or restart it (if it was killed).

NOTE: Depending on the capabilities of the Host device, for example the number of decoders or the available processing power and memory, the Host may decide to either push the video/broadcast object to the background while the focus is on the AV Control object, and subsequently pull it to the foreground again, or alternatively kill the video/broadcast object and restart it at a later stage. Although such a choice is likely to have an effect on the perceived user experience, it is mainly a performance issue and therefore left up to the implementation.

9.2.3 Local PVR

The standardized STB-less IPTV system described in this document does not explicitly support local PVR functionality.

9.2.4 Local timeshift

This document does not mandate the Host to contain storage to support local timeshift, and the local timeshift functionality as described in this section is OPTIONAL. However, if the Host supports local timeshift for scheduled content services, it SHALL make this functionality available to the IPTV App in the manner described by this section.

In order to allow the IPTV App to check whether the Host supports local timeshift, the Host SHALL implement the xmlCapabilities property specified in section 7.15.3 of [OIPF_DAE2]. Once the xmlCapabilities property of the Host is queried, it SHALL respond with a capabilities XML in accordance with section 9.3 and Annex F of [OIPF_DAE2]. The capabilities XML SHALL at least include an element of the type 'videoBroadcastType' with a 'localTimeshift' attribute.

NOTE: Within the context of this document, the capabilities XML is only used to convey the presence of local timeshift functionality in the Host. As such, the IPTV App MAY ignore the rest of the capabilities XML.

If the 'localTimeshift' attribute in the capabilities XML provided by the Host indicates it supports localTimeshift, the Host SHALL support the constants, properties and methods specified in section 7.13.2 of [OIPF_DAE2], with the possible exception of the recordNow() and stopRecording() methods. The recordNow() and stopRecording() methods MAY be supported, but are not used by any features specified in this document.

If at any point while consuming a scheduled content service the user presses one of the remote control buttons typically associated with timeshift functionality, and the IPTV App has confirmed that local timeshift is supported on the Host, the IPTV App MAY use the timeshift APIs from section 7.13.2 of [OIPF_DAE2] to control playback in the Host. Alternatively, the IPTV App MAY decide to use a network-based timeshift service instead, if such a service is available (for an example of such functionality see section 9.2.2).

 Copyright 2014 © Open IPTV Forum e.V.