Application Discovery over Broadband Specification

Contents

Introduction

The versions of TS 102 796 [1] published to date rely on signalling in the broadcast to start broadcast-related applications, through the Application Information Table (AIT). The present document defines methods for discovery of broadcast-related HbbTV® services via a broadband internet connection for circumstances when the AIT and related signalling via the broadcast network is not available to the HbbTV® terminal. The discovery methods rely on retrieving or extracting a unique identifier for a broadcast channel and then starting a discovery process to find a server that can be contacted to retrieve an AIT over the broadband connection. The broadband-retrieved AIT would only be used if no AIT is available in the broadcast channel. The discovery method relies on the Internet's DNS system. In simplified form, the process works as follows:

- Extract a unique identifier from the broadcast channel.
- With the unique identifier, perform a DNS query to find (resolve) the AIT server.
- Ask the AIT server for an AIT that matches the broadcast channel.
- Using the AIT, retrieve the HbbTV® application.

The present document is targeted at two main deployment scenarios:

- HbbTV® terminals connected to a DVB network which does not carry the HbbTV® AIT. In this case the unique identifier is based on DVB Service Information.
- HbbTV® TV sets connected via HDMI to a STB that is in turn connected to the DVB network. In this case the unique identifier is based on information carried in the video and audio content (referred to as a 'watermark' in the present document). This method has the additional capability of enabling discovery of a media timeline and stream events. This capability can also be employed when a terminal is connected to a DVB network that does not carry timeline or stream events.

Both discovery methods can also be used in other deployment scenarios, for example the watermark can be used for application discovery from a DVB broadcast and the service information approach could be adapted for use with IPTV or live OTT solutions using proprietary service discovery.

An AIT retrieved over the broadband connection and the Application referenced in that AIT are not necessarily the same as the AIT that would be available in the broadcast and the associated HbbTV® Application. Generally, when no AIT is available in the broadcast, then neither would be event signalling, and the provider of the application may want to resort to alternative methods for providing event signalling to the HbbTV® application. When discovery using DVB is employed, the application may have to be modified to receive events in another manner (e.g. via broadband). When discovery using watermarking is employed, stream events may be delivered via the watermark.

The discovery method that employs DVB Service Information does not allow for application changes over time - e.g. when the program changes. When this method is employed, the application will have to include the necessary logic. Entities relying on the functionality provided in the present document are advised to consider these limitations when writing their applications.

1 Scope

The present document augments clause 6 of TS 102 796 [1], which states that broadcast-related applications are signalled as part of the broadcast. It defines a method for discovery of HbbTV® applications in settings where AIT signalling via the broadcast network is not available to the terminal. In this situation, an HbbTV® terminal may discover broadcast-related HbbTV® services via a broadband internet connection.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference.

The following referenced documents are necessary for the application of the present document.

[1] TS 102 796: "Hybrid Broadcast Broadband TV".

NOTE: Including the latest errata as published on http://hbbtv.org/resource-library/#specifications.

[2] TS 102 796 (V1.4.1): "Hybrid Broadcast Broadband TV".

NOTE: Including the latest errata as published on http://hbbtv.org/resource-library/#specifications.

[3] EN 300 468: "Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB systems".

[4] Open IPTV Forum Release 2 specification, volume 5 (V2.3): "Declarative Application Environment".

[5] TS 102 034 (V1.5.1): "Digital Video Broadcasting (DVB); Transport of MPEG-2 TS Based DVB Services over IP Based Networks".

[6] TS 102 809: "Digital Video Broadcasting (DVB); Signalling and carriage of interactive applications and services in Hybrid Broadcast/Broadband environments".

[7] IETF RFC 6066: "Transport Layer Security (TLS) Extensions: Extension Definitions".

[8] W3C Recommendation (Second Edition) (10 June 2008): "XML Signature Syntax and Processing".

[9] ATSC A/336: "Content Recovery in Redistribution Scenarios".

[10] IETF RFC 1034: "Domain Names - Concepts and Facilities".

[11] IETF RFC 1035: "Domain Names - Implementation and Specification".

[12] ATSC A/334: "Audio Watermark Emission".

[13] ATSC A/335: "Video Watermark Emission".

[14] ISO/IEC 8859-5: Information technology -- 8-bit single-byte coded graphic character sets -- Part 5: Latin/Cyrillic alphabet".

[15] TS 103 736-1: "Hybrid Broadcast Broadband TV; Targeted Advertising; Part 1: Functional requirements"

[16] TS 103 736-2: "Hybrid Broadcast Broadband TV; Targeted Advertising; Part 2: Non-functional requirements"

[17] W3C Recommendation (15 December 2016) "WebIDL Level 1"

NOTE: Available at https://www.w3.org/TR/WebIDL-1/

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] TS 103 270 (V1.1.1): "RadioDNS Hybrid Radio; Hybrid lookup for radio services".

[i.2] TS 103 286-2 (V1.2.1):"Digital Video Broadcasting (DVB); Companion Screens and Streams; Part 2: Content Identification and Media Synchronization".

[i.3] ISO/IEC 13818-1:2018: "Information technology -- Generic coding of moving pictures and associated audio information -- Part 1: Systems".

[i.4] IETF RFC 4033: "DNS Security Introduction and Requirements".

[i.5] IETF RFC 4034: "Resource Records for the DNS Security Extensions".

[i.6] IETF RFC 4035: "Protocol Modifications for the DNS Security Extensions".

[i.7] TS 103 555: "IP-delivered Broadcast Channels and Related Signalling of HbbTV Applications".

[i.8] TS 103 606: "Hybrid Broadcast Broadband Television; Operator Applications"

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in TS 102 796 [1] and the following apply:

AIT server: server that for provides broadcast-related AIT(s) over broadband

audio watermark segment: VP1 audio watermark segment as defined in clause 5.2.5 of ATSC A/336 [9]

authoritative FQDN: internet domain for a (HbbTV®) service provider

discovered AIT: broadcast related AIT retrieved according to the present document

HbbTV® DNS FQDN: internet domain constructed only for the purpose of querying DNS

interval_field: field in the ATSC A/336 VP1 payload containing the interval code

interval code: value that identifies the interval of content in which the VP1 payload value is embedded

query flag: value of the query_flag field in an instance of the VP1 Payload as defined in ATSC A/336 [9]

server code: value that identifies a server which acts as the starting point for acquisition of supplementary content

server field: field in the ATSC A/336 VP1 payload containing the server code

video watermark segment: VP1 video watermark segment as defined in clause 5.1.7 of ATSC 336 [9]

watermark segment: audio watermark segment or a video watermark segment

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in TS 102 796 [1] and the following apply:

STB	Set-Top Box
TTL	Time To Live

4 Overview (informative)

The methodology is modelled after RadioDNS [i.1], using certain parameters provided in the broadcast for the identification of services. These could be digital parameters extracted from DVB Service Information, e.g. DVB-SI [3], or parameters encoded in a digital watermark that is inserted for the purpose of enabling this discovery method. The present document defines operation either in the presence of DVB Service Information or, alternatively, in the presence of specified audio and/or video watermarks; other operation modes may be added in a future version.

The discovery method allows broadcasters to uniquely associate an AIT server with their channel and comprises discovering an authoritative FQDN for an AIT server, using DNS queries to hbbtvdns.org, a root domain name server.

NOTE: It is possible for local markets to define a market-specific alternative to hbbtvdns.org; this is not further addressed in the present document.

The protocol for discovery and retrieval of an AIT for a single service follows a number of steps, as outlined below. Figure 1 illustrates the steps using the example of DVB-based parameters and figure 2 illustrates the steps using the example of watermark-based parameters:

- The terminal queries a DNS recursive resolver using an HbbTV® DNS FQDN constructed from information present in the broadcast (1). A DNS recursive resolver known to the terminal returns the authoritative FQDN for that service (2), acquiring the mapping (if available) from the root domain name server if it is not locally cached.
- Either (1) When the terminal is receiving a DVB broadcast, there is no AIT in the broadcast signal or (2) When the terminal is presenting video from HDMI and receives a watermark as defined in the present document; then the terminal retrieves an XML-encoded AIT from the server using a URL constructed from authoritative FQDN (3 and 4). AIT retrieval includes acquisition of the AIT server address by the terminal from a DNS recursive resolver, which in turn acquires it (if available) from the broadcaster's Authoritative FQDN DNS root if it is not locally cached.
- The terminal uses the AIT for that broadcast service and retrieves the application (5 and 6).

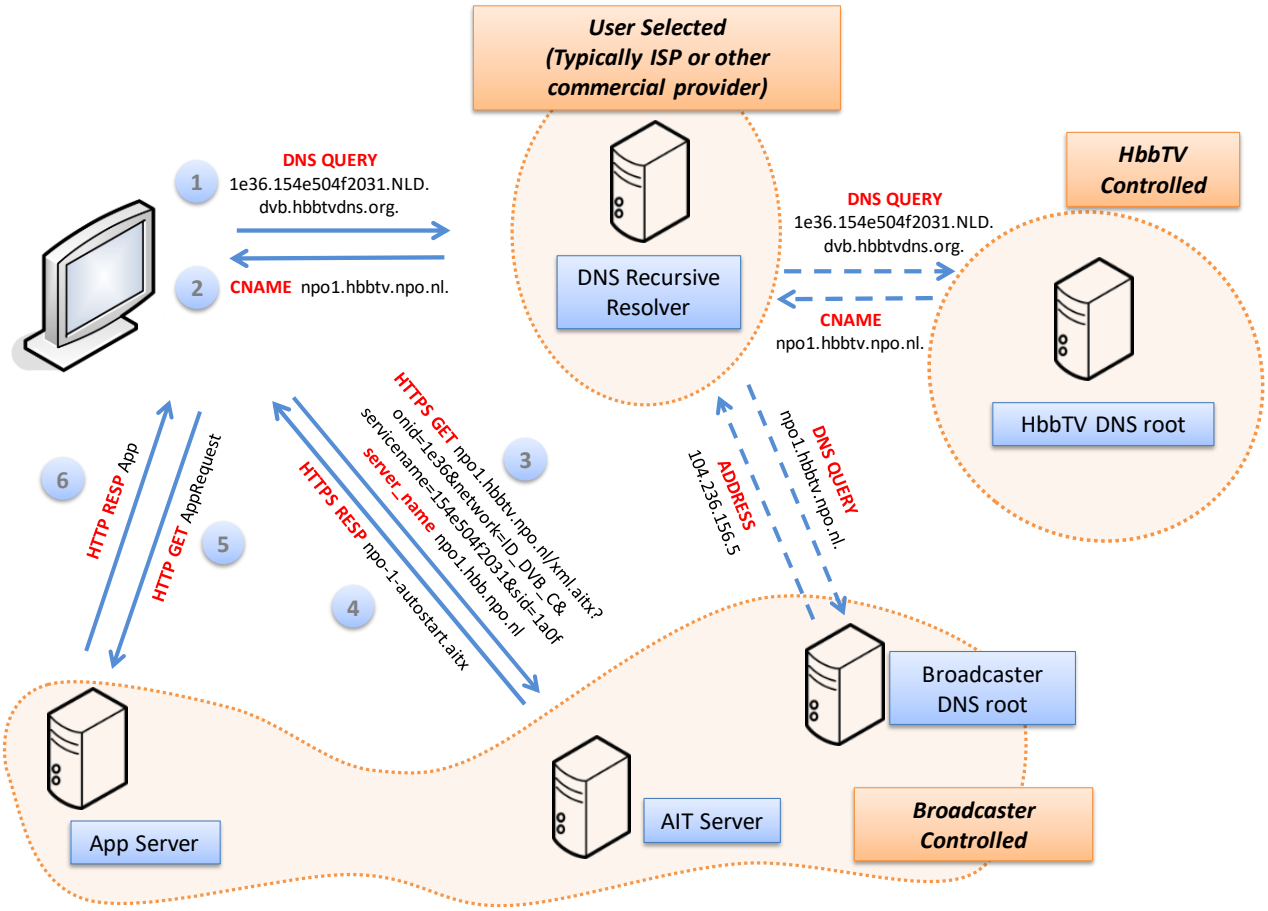

Figure 1: HbbTV® Application Discovery over Broadband in the presence of DVB Service Information

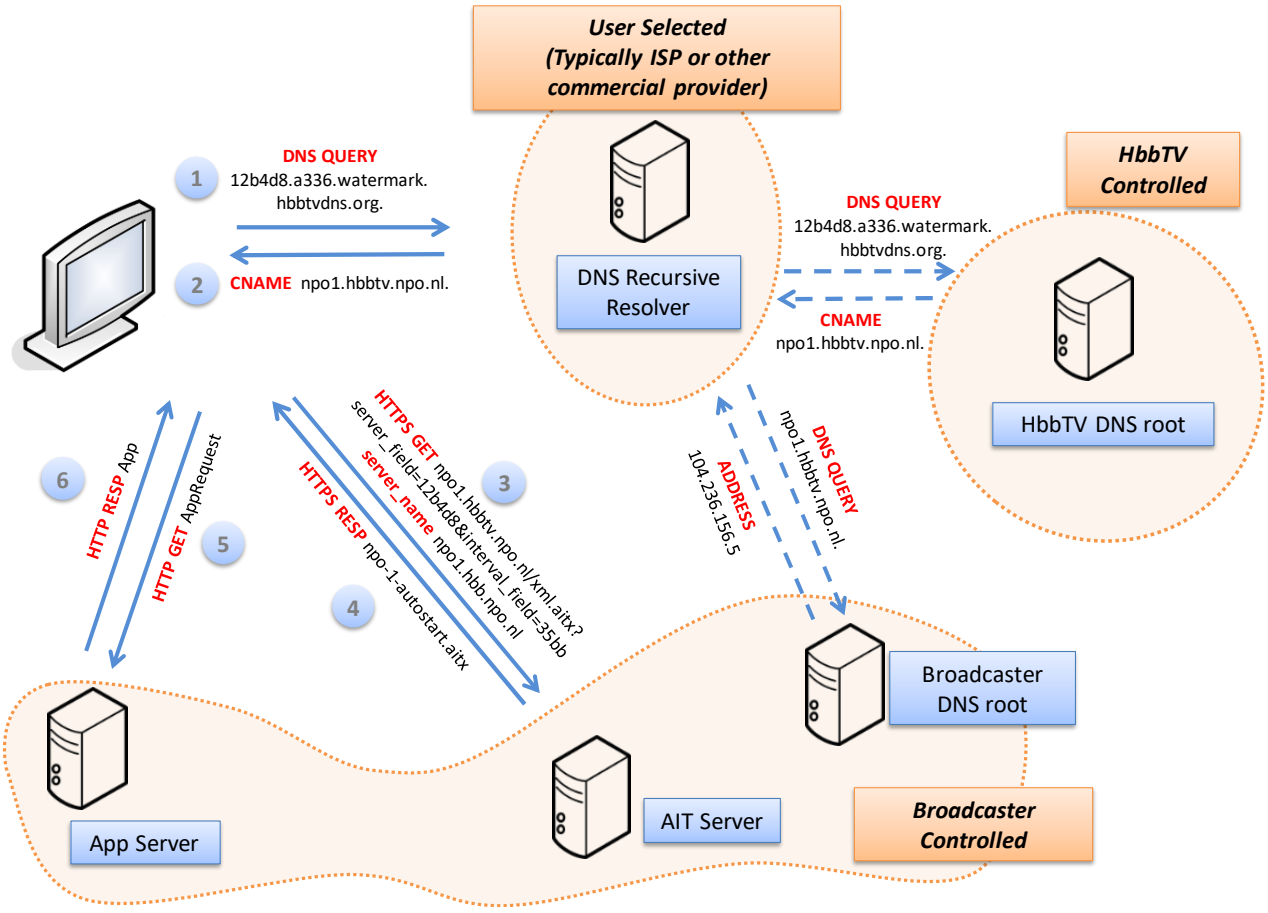

Figure 2: HbbTV® Application Discovery over Broadband in the presence of watermark information

Application discovery over broadband as defined in the present document may be used in combination with targeted advertising as defined in TS 103 736-1 [15].

- When application discovery over broadband is used with services received via DVB transmission, TS 103 736 -1 [15] and TS 103 736-2 [16] are applied as-is.
- When application discovery over broadband is used with services received via HDMI, additional requirements apply that are defined in the present document. See clauses 8.2, 9.6, 10.2.3, A.2.2, and A.3.

It is optional for terminals supporting TS 103 736-1 [15] and also supporting the present document to support the use of the two in combination.

5 HbbTV® Application Discovery over Broadband

5.1 Introduction

This clause defines a method for discovery and signalling of broadcast-related applications, which are discovered and signalled via broadband, instead of being signalled as part of the broadcast channel as defined in clause 7.2.3.1 of TS 102 796 [1].

The structure of this clause reflects the fact that the present document defines generic methods as well as instantiations that map on specific ways in which identifiers can be extracted and used. The present version defines such a mapping for a terminal that receives a DVB signal and one that relies on watermarking, to enable operation of terminals that do not receive the digital broadcast signal, for instance when the terminal is connected to a set top box with an HDMI cable. Mappings for non-DVB digital broadcasts can be added based on the HbbTV® IPTV specification TS 103 555 [i.7] or platform specific integration.

Terminals shall implement all of the mandatory requirements in the TS 102 796 [1], except where explicitly stated otherwise in the present document.

5.2 Discovering broadcaster AIT servers

The terminal shall attempt to discover broadcasters' AIT servers according to the process as described below. This process is independent from service selection by the user and shall be executed in the following cases:

- For each service in the terminal's channel list (see TS 102 796 [1] and the OIPF DAE specification [4]) and for each server_field in the server field cache (if application discovery using watermarking is supported), when the terminal is powered on. These attempts shall be made in alphabetical order by HbbTV® DNS FQDN.
- For any service where the terminal detects a change in the service name.
- For any service that is added to the terminal's channel list.
- For every service in the terminal's channel list, when the terminal's country setting is changed.
- For every server field value, when it is added to the server field cache.

Discovery of an AIT server shall be performed in the following way:

- The Authoritative FQDN shall be resolved as specified in clause 5.5, using Service Identification as specified in clause 5.3 and the HbbTV® DNS FQDN construction as specified in clause 5.4.

The following caching rules shall apply to DNS resolution performed for resolution of the Authoritative FQDN as specified in clause 5.5:

- DNS resource records shall be cached by the terminal in accordance with the resolver caching rules of IETF RFC 1034 [10] and IETF RFC 1035 [11], as amended by the present clause.
- Cached DNS resource records shall not be retained over a power cycle.
- Terminals shall be capable of simultaneously caching the DNS resource records of all services in the channel list and, if the terminal supports application discovery using watermarking, all server fields in the server_field cache as defined below.
- If a DNS resource record retrieval returns a name error (i.e. the record does not exist), the terminal shall cache this negative response with a TTL of 24 hours.
- Terminals shall refresh each cached DNS resource record once it has been stored in the cache for a number of seconds equal to the TTL associated with the DNS record (as defined in clause 5.1 of IETF RFC 1035 [11]), independently of and asynchronous to AIT retrieval.

NOTE: It is understood that terminals typically incorporate a DNS stub resolver that does not perform caching and rely on a remote recursive resolver identified via DHCP for caching. Terminals are not expected to implement a recursive resolver for the purpose of complying with these requirements. The DNS caching behaviour is expected to be included as part of a terminal implementation that employs a stub resolver.

The server field cache enables the terminal to populate its DNS cache with the records associated with previously viewed watermarked services upon power-up. Terminals supporting application discovery based on watermarking shall apply the following caching rules to server_field values detected from watermarks:

- server_field values detected from watermarks shall be cached by the terminal (the "server field cache").
- The server field cache shall be retained across power cycles and erased only upon user request (e.g. via a terminal feature such as "restore factory settings" or "delete stored information"). To ensure that the cache is

retained when power is removed from the terminal entirely, terminals shall write changes to server field cache data to persistent storage within 5 minutes of the terminal being put into standby and should write changes to server field cache data to persistent storage soon after that data has been set or modified, e.g. within 30 s.

- Terminals shall be capable of storing 200 server_field values in the server field cache and the cached values shall not expire. However, if the cache does not have space to store a new server_field value, it shall replace the oldest (i.e. least recently added) entry in the cache.

5.3 Service Identification

5.3.1 Service Identification in the presence of DVB Service Information

For terminals supporting application discovery over broadband using DVB Service Information, identification of a service shall be provided by a combination of DVB service parameters. The parameters are defined in table 1.

Table 1: HbbTVDNS parameters

Parameter	Description	Value
Country	3-character country code as specified by the `Configuration.countryId` property in Open IPTV Forum Release 2 specification [4].	3-char string
Servicename	The DVB service name bytes from the service_name field in the service_descriptor in the SDT for this service as defined in EN 300 468 [3], encoded as a string of two digit hex bytes.	Up to 256 two digit hex bytes
Onid	The original network id for this service as defined in EN 300 468 [3] encoded as a string of two digit hex bytes.	4-digit lower case hex string

In the case of using the present document in the context of IPTV or OTT networks, mappings of DVB-SI and equivalent information can be done based on the HbbTV® IPTV specification [i.7] or via platform specific integration.

5.3.2 Service Identification using ATSC 3 watermarks

For watermarks whose payload is according to clause 5.2.3 of ATSC A/336 [9], services are identified using the server_field (Server Code) from the vp1_payload structure.

5.4 HbbTV® DNS FQDN Construction

5.4.1 HbbTV® DNS FQDN Construction in the presence of DVB Service Information

The terminal shall construct a HbbTV® DNS FQDN as follows:

```
<onid>.<servicename>.<country>.dvb.hbbtvdns.org
```

The terminal shall extract and re-encode all the bytes from the service_name field in the service_descriptor in the SDT as transmitted including any optional character selection byte. No character set processing or translation shall be performed. This shall be done regardless of what character sets a terminal supports.

Some examples of HbbTV® DNS FQDNs constructed from broadcast parameters are shown in table 2.

Table 2: Examples of HbbTV® DNS FQDN construction for HbbTV®

Country	Servicename	Onid	HbbTV® DNS FQDN
NLD	154e504f2031 ("NPO 1" with the 0x15 indicating UTF-8 character encoding as defined by EN 300 468 [3])	1e36	1e36.154e504f2031.NLD.dvb.hbbtvdns.org
DEU	10415244 ("ARD" with the 0x10 indicating ISO/IEC 8859-5 [14] character encoding as defined by EN 300 468 [3]).	2345	2345.10415244.DEU.dvb.hbbtvdns.org

5.4.2 HbbTV® DNS FQDN Construction using ATSC 3 watermarks

For watermarks whose payload is according to clause 5.2.3 of ATSC A/336 [9], the terminal shall construct an HbbTV® DNS FQDN as follows;

```
<server_field>.a336.watermark.hbbtvdns.org
```

The server_field from the most recently detected audio watermark shall be used, except in the Verified video watermark detected only state (see table 5) where the server_field from the most recently detected video watermark payload shall be used. The bits of the server_field shall be encoded as a hexadecimal number without leading zeros and with 10 to 15 being represented by lower case 'a' through 'f'.

EXAMPLE: A server code of binary 001 0010 1011 0100 1101 1000 is encoded as 12b4d8

Only the large_domain syntax is required to be supported.

5.5 Resolution of Authoritative FQDN

The terminal shall perform a DNS request using the HbbTV® DNS FQDN, to acquire the Authoritative FQDN. The response to this request contains a single CNAME record [11] containing the Authoritative FQDN of the service provider. If no CNAME is returned (a "negative response"), then a broadband-discoverable AIT service has not been registered.

EXAMPLE: Consider a TV service identified by the HbbTV® DNS FQDN:

```
1e36.154e504f2031.NLD.dvb.hbbtvdns.org
```

A DNS lookup will yield the following lookup result:

```
1e36.154e504f2031.NLD.dvb.hbbtvdns.org. 86400    IN CNAME npo1.hbbtv.npo.nl.
```

Therefore, for this service, the Authoritative FQDN is:

```
npo1.hbbtv.npo.nl
```

5.6 AIT retrieval

5.6.1 AIT retrieval in the presence of DVB Service Information

The terminal shall retrieve the AIT by performing an HTTPS GET where the TLS client connection request includes the Authoritative FQDN in the Server Name Indication (SNI) as specified in IETF RFC 6066 [7] request to the following address:

```
https://<domain-name>/xml.aitx?onid=<onid>&network=<network>&servicename=<servicename>&sid=<sid>
```

The parameters that the terminal shall provide are shown in table 3.

Table 3: AIT request parameters

Parameter	Description	Value
domain-name	The Authoritative FQDN obtained as specified in clause 5.5 and cached as specified in clause 5.2.	String
onid	Original network ID, as provided in the SDT from DVB-SI, EN 300 468 [3].	4-digit lower case hex string
network	Identifies the broadcast delivery system. Values correspond to the permitted values for the `idType` property of the Channel class as defined in clause 7.13.11 of the OIPF DAE specification [4].	String
servicename	See table 1.	See table 1
sid	Service ID, as used in DVB-SI, EN 300 468 [3].	4-digit lower case hex string

EXAMPLE: Consider a DNS record as provided by DNS:

```
1e36.154e504f2031.NLD.dvb.hbbtvdns.org. 86400    IN CNAME npo1.hbbtv.npo.nl.
```

The URL to retrieve the AIT for this service is:

```
https://npo1.hbbtv.npo.nl/xml.aitx?onid=1e36&network=ID_DVB_C& servicename=154e504f2031&sid=1a0f
```

5.6.2 AIT retrieval when ATSC 3 watermarks are used

The terminal shall retrieve the AIT by performing an HTTPS GET request to the following address:

```
https://<domain-name>/xml.aitx?server_field=<server_field>&interval_field=<interval_field>
```

The <domain-name> shall be the Authoritative FQDN obtained as specified in clause 5.5 and cached as specified in clause 5.2. The server_field and interval_field from the most recently detected audio watermark payload shall be used, except in the Verified video watermark detected only state (see table 5) where the server_field and interval_field from the most recently detected video watermark payload shall be used. The bits of the server_field and interval_field shall be encoded as hexadecimal numbers without leading zeros and with 10 to 15 being represented by lower case 'a' through 'f'.

EXAMPLE: A server code of binary 001 0010 1011 0100 1101 1000 is encoded as 12b4d8

5.6.3 Common AIT retrieval requirements

When performing AIT retrieval according to clauses 5.6.1 or 5.6.2, the following requirements shall apply:

1) The requirements of clause 7.3.2.5 of TS 102 796 (V1.4.1) [2] shall apply to AIT retrieval regardless of what is TS 102 796 [1].

2) The response to this request shall have the following MIME type: application/vnd.dvb.ait+xml and an XML AIT as payload. The AIT is contained in a single application discovery record.

6 Service and application model

6.1 Introduction

This clause defines procedure for use by terminals to perform application discovery, signalling, and lifecycle management of applications via broadband. These procedures include:

- Priority between application discovery from broadcast AITs, DVB Service Information, and watermarks in scenarios where multiple of these mechanisms is applicable (clause 6.2);
- Application lifecycle management when application discovery using DVB Service Information is employed (clause 6.2.1);
- Monitoring for watermarks and a state machine for watermarking processing (clause 6.3);

- Application lifecycle management when application discovery using watermarking is employed (clause 6.4).

6.2 Priority between application discovery mechanisms

6.2.1 Services received via DVB transmission

Broadcast AIT reception as defined in TS 102 796 [1], application discovery over broadband using DVB Service Information and application discovery over broadband using watermarks (both the latter as defined in the present document) are all applicable techniques for application discovery. When more than one of these techniques is supported by the terminal, more than one technique may be initiated simultaneously for application discovery.

When a terminal that employs application discovery over broadband changes to a service, if an AIT server is cached for that service, the terminal shall attempt to retrieve the Discovered AIT as defined in clause 5.6. If a Discovered AIT has been obtained for the current service, the terminal shall decide whether or not to use it according to the following policy:

1) If the PMT of the broadcast service references a PID carrying a valid application_signalling_descriptor, and an AIT section for the HbbTV® application_type is receivable from that PID within 30 s, the AIT provided in the broadcast service shall be used.

2) If the PMT of the broadcast services does not reference a PID carrying a valid application_signalling_descriptor, the Discovered AIT shall be used immediately.

3) If the PMT of the broadcast service references a PID carrying a valid application_signalling_descriptor, but an AIT section for the HbbTV® application_type has not been observed after continuously monitoring the AIT PID for 30 s, the Discovered AIT shall be used.

4) Relying on implementation-dependent logic, a terminal may use a Discovered AIT sooner than indicated in steps 1) and 3) above, for instance when no AIT has been observed on that particular service in the recent past.

If a DNS negative response for the AIT server needed for application discovery via broadband is cached for the current service then the terminal shall treat this identically to a regular broadcast service that does not include an AIT. The terminal shall not attempt to discover a AIT server under these conditions.

The terminal shall apply the lifecycle rules defined in clauses 6.2.2.2 and 6.2.2.3 of TS 102 796 [1] for broadcast-related applications where all references to AIT shall be replaced by Discovered AIT.

If a terminal connected to a DVB network not carrying the HbbTV® AIT supports application discovery over broadband both 1) using DVB Service Information and 2) using watermarks then the terminal shall attempt both forms of discovery in parallel. The first Discovered AIT obtained by the terminal shall be used as the basis for the HbbTV® application lifecycle. The second (slower) discovery process shall either be stopped or any eventual result silently ignored.

Once an application is launched and running, the terminal shall continue to employ the same technique that was used to acquire the AIT from which the application was launched for lifecycle management of that application until execution of that application is stopped or becomes broadcast independent, regardless of whether an alternate technique for application discovery is (or becomes) available. Subsequently, the terminal may again employ more than one of the specified techniques for application discovery until an application is again launched.

6.2.2 Services received via HDMI

The only applicable application discovery mechanism for services received via HDMI is application discovery over broadband using watermarks (as defined in the present document).

6.2.3 Services received via other input sources

Application discovery over broadband using watermarks (as defined in the present document) may also be applicable to non-DVB input sources such as IPTV or OTT.

6.3 Watermark states and transitions

6.3.1 State machine

Table 4 defines the states and state transitions for watermark detection and how this maps to the HbbTV® application discovery process. States are defined as follows:

1) No watermark detected (initial state)

2) Unverified Video watermark detected only

3) Verified Video watermark detected only

4) Audio watermark detected only

5) Audio and verified video watermarks detected

6) Audio and unverified video watermarks detected

NOTE 1: See clause 6.3.2 below for a definition of what it means for the video watermark to be verified or not.

Transitions are defined from each state based on a number of events that can happen and trigger a change. Actions are defined for transitions where appropriate based on the following conditions happening:

- The terminal detects the start of a video watermark segment (see table 5).
- The terminal detects the start of an audio watermark segment (see table 5).
- The terminal detects the end of an audio watermark segment (i.e. complete loss of audio watermark, change in server code, discontinuity in interval code) (see table 6).
- The terminal detects the end of a video watermark segment (i.e. complete loss of video watermark, change in server code, discontinuity in interval code) (see table 7).
- An update to the XML-AIT corresponding to the currently received watermark results in a state change (e.g. previously unverified video watermark becoming verified or vice-versa) (see table 9).

When either of the following happens, running broadcast-related applications controlled by the watermark shall be stopped:

- the user selects a different input or source for the terminal (e.g. changing from an HDMI input to Smart TV); or
- the input over which the watermark has been received is lost (e.g. the user turns off the set-top box connected to an HDMI input).

In the case of change in server code or discontinuity in interval code, the transition(s) resulting from the end of one Watermark Segment shall be processed before the transition(s) resulting from the start of the next watermark Segment.

NOTE 2: Truly simultaneous detection or loss of video and audio watermarks is not included in this state machine. One is detected before the other and the events then processed sequentially.

In the case of change in query flag, there is no transition between states however a terminal action may be required depending on the current state (see table 8).

A watermark state diagram including the transitions described in tables 1 to 6 is shown in figure 3.

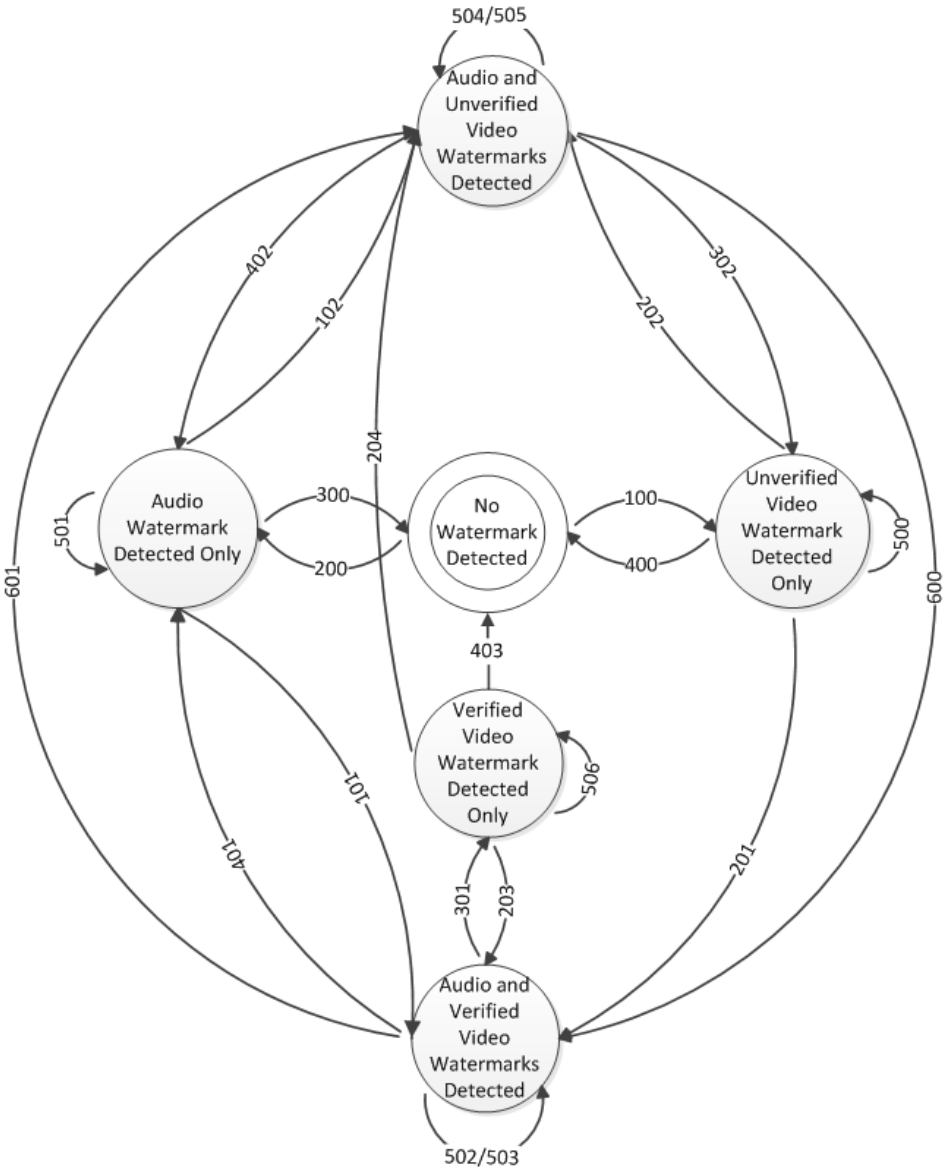

Figure 3: Watermark state diagram

Table 4: State transitions when the terminal detects the start of a Video Watermark Segment

	Initial State	Trigger	New state	Typical Causes (informative)	Action
100	No watermark detected	Video watermark detected	Unverified video watermark detected only	1) Content change and video watermark is detected before audio watermark 2) Content change while audio is muted on STB 3) Video watermark has been inserted by an attacker 4) Content change and audio watermark has been removed by an attacker	No action. Unverified video watermarks shall not be used for application discovery.
101	Audio watermark detected only	Video watermark detected that is verified using the audio watermark (see clause 6.3.2)	Audio and Verified Video watermarks detected	1) User dismisses UI on STB which was scaling the video or obscuring the top line of the video 2) Content change and audio watermark was detected before video watermark 3) Video component selection is changed on the STBSTB 4) Attacker has stopped removing video watermark	No action. Continue managing the HbbTV® application lifecycle using audio watermark.
102	Audio watermark detected only	Video watermark detected that is not verified using the audio watermark	Audio and Unverified Video watermarks detected	1) Content change and new video watermark has been detected prior to detecting change in audio watermark 2) Error by the broadcaster 3) Video watermark has been inserted by an attacker	No action. Continue managing the HbbTV® application lifecycle using audio watermark. Unverified video watermarks shall not be used for application lifecycle.
	Verified video watermark detected only	Not applicable			
	Audio and verified video watermarks detected	Not applicable			
	Audio and unverified video watermarks detected	Not applicable			
	Unverified video watermark detected	Not applicable			

Table 5: State transitions when the terminal detects the start of an Audio Watermark Segment

	Watermark State	Trigger	New state	Typical Causes (informative)	Action
200	No watermark detected	Audio watermark detected	Audio watermark detected only	1) Content change and audio watermark is detected before video watermark 2) Content change to content where video watermark is not used 3) Content change while STB is scaling the video or obscuring the top line of the video	Start HbbTV® application discovery using data from audio watermark (see clause 6.4.2).
201	Unverified video watermark detected only	Audio watermark is detected that verifies the video watermark (see clause 6.3.2)	Audio and verified Video watermarks detected	1) Content change has occurred and video watermark was detected before audio watermark 2) Content change has occurred while audio was muted on STB and unmute has now occurred	Start HbbTV® application discovery using data from audio watermark (see clause 6.4.2).
202	Unverified video watermark detected only	Audio watermark detected that does not verify the video watermark (see clause 6.3.2)	Audio and unverified Video watermarks detected	1) Multiple channel changes occurred and audio watermark is now detected on current channel after a video watermark was detected on a previous channel but before audio watermark was detected on the previous channel 2) Error by the broadcaster 3) Video watermark has been inserted by an attacker	Start HbbTV® application discovery using data from audio watermark (see clause 6.4.2). Unverified video watermarks shall not be used for the application discovery process.
203	Verified video watermark detected only	Audio watermark detected, audio watermark still verifies the video watermark (see clause 6.3.2)	Audio and verified Video watermarks detected	1) Audio is unmuted on STB 2) Audio component selection is changed on STB	No action. Continue managing the HbbTV® application lifecycle (see clause 6.4.2).
204	Verified video watermark detected only	Audio watermark detected that does not verify the video watermark (see clause 6.3.2)	Audio and unverified video watermarks detected	Content change has occurred while audio was muted on STB	Start application discovery using data from audio watermark (see clause 6.4.2).Unverified video watermarks shall not be used for the application discovery.
	Audio watermark detected only	Not applicable			
	Audio and verified video watermarks detected	Not applicable			
	Audio and unverified video watermarks detected	Not applicable			

Table 6: State transitions when the terminal detects the end of an Audio Watermark Segment

	Watermark State	Trigger	New state	Typical Causes (informative)	Action
	No watermark detected	Not applicable			
300	Audio watermark detected only	Audio watermark lost	No watermark detected	1) Content change has occurred and loss of video watermark was detected before loss of audio watermark 2) Content change has occurred for content using audio watermark only 3) Content change has occurred while STB is scaling the video or obscuring the top line of the video 4) Audio muted on the STB while STB is scaling the video or obscuring the top line of the video 5) Audio muted on the STB for channel using audio watermark only	Start process for loss of watermark (see clause 6.4.3).
	Video watermark detected only	Not applicable			
301	Audio and verified video watermarks detected	Audio watermark lost	Verified video watermark detected only	1) Content change has occurred and loss of audio watermark is detected before loss of video watermark 2) Audio component selection is changed on STB and audio watermark has not yet been detected on the new component 3) Audio is muted on the STB	No action. Continue managing the HbbTV® application lifecycle using video watermark. The video watermark is obeyed because it was verified by the last audio watermark detected.
302	Audio and unverified video watermarks detected	Audio watermark lost	Unverified video watermark detected only	1) Channel change has occurred and loss of audio watermark from prior channel is detected after video watermark from new channel is detected 2) Audio mute on the STB during attack 3) Audio mute on the STB during error by broadcaster	Start process for loss of watermark (see clause 6.4.3). The video watermark shall be ignored because it was not verified by the last audio watermark detected.

Table 7: State transitions when the terminal detects the end of a Video Watermark Segment

	Watermark State	Trigger	New state	Example (informative)	Action
	No watermark detected	Not applicable			
	Audio watermark detected only	Not applicable			
400	Unverified video watermark detected only	Video watermark lost	No watermark detected	1) Multiple channel changes occurred and video watermark on prior channel was detected before audio watermark on prior channel was detected 2) Error by broadcaster ends 3) Attack on the system ends	No action. Application discovery is not supported by unverified video watermark.
401	Audio and verified Video watermarks detected	Video watermark lost	Audio watermark detected only	1) Content change has occurred and loss of video watermark is detected before loss of audio watermark 2) STB begins scaling the video or obscuring the top line of the video 3) Video component selection is changed on STB	No action. Continue managing the HbbTV® application lifecycle using audio watermark.
402	Audio and unverified video watermarks detected	Video watermark lost	Audio watermark detected only	1) Multiple channel changes occurred and video watermark is detected and lost a prior channel before loss of audio watermark on an earlier prior channel 2) Error by the broadcaster ends 3) Attack on the system ends	No action. Continue supporting application discovery using audio watermark.
403	Verified video watermark detected only	Video watermark lost	No watermark detected	1) Content change has occurred and loss of video watermark is detected after loss of audio watermark 2) Content change while audio is muted on STB 3) Video component selection is changed while audio is muted on STB 4) STB begins scaling the video or obscuring the top line of the video while audio is muted on STB	Start process for loss of watermark (see clause 6.4.3)

Table 8: Events without state transitions

	Watermark State	Trigger	Typical Causes (informative)	Action
500	Unverified video watermark detected only	State change of Query Flag is detected in video watermark	1) Content change and AIT update is requested by broadcaster before audio watermark is detected 2) Video watermark has been inserted by an attacker 3) Audio watermark has been removed by an attacker	No action. Unverified video watermarks shall not be used for application discovery.
501	Audio watermark detected only	State change of Query Flag is detected in audio watermark	1) AIT update is requested by broadcaster	Acquire and process AIT using data from the audio watermark (see clause 6.4.2.2).
502	Audio and verified video watermark detected	State change of Query Flag is detected in video watermark	1) AIT update is requested by broadcaster	Acquire and process AIT using data from the audio watermark (see clause 6.4.2.2).
503	Audio and verified video watermark detected	State change of Query Flag is detected in audio watermark	1) AIT update is requested by broadcaster	Acquire and process AIT using data from the audio watermark (see clause 6.4.2.2).
504	Audio and unverified video watermark detected	State change of Query Flag is detected in audio watermark	1) AIT update is requested by broadcaster	Acquire and process AIT using data from the audio watermark (see clause 6.4.2.2).
505	Audio and unverified video watermark detected	State change of Query Flag is detected in video watermark	1) Channel change has occurred and AIT update is requested by broadcaster on new channel before loss of audio watermark is detected on prior channel 2) Channel change has occurred and broadcaster AIT update request is detected on prior channel before loss of watermark is detected 3) Broadcaster error 4) Video watermark is inserted by an attacker	No action. Unverified video watermarks shall not be used for application discovery.
506	Verified video watermark detected only	State change of Query Flag is detected in video watermark	1) AIT update is requested by broadcaster	Acquire and process AIT using data from the video watermark (see clause 6.4.2.2)

Table 9: State transitions based on AIT changes

	Watermark State	Trigger	New state	Example (informative)	Action
600	Audio and unverified video watermarks detected	Discovered AIT is obtained that verifies the video watermark	Audio and verified video watermarks detected	Discovered AIT is obtained after both audio and video watermarks are detected	No action. Continue managing the HbbTV® application lifecycle using audio watermark.
601	Audio and verified video watermarks detected	AIT update is obtained that no longer verifies the video watermark	Audio and unverified video watermarks detected	Broadcaster publishes an updated AIT that no longer contains the component element that previously verified the detected video watermark	No action. Continue managing the HbbTV® application lifecycle using audio watermark.

6.3.2 Verification of video watermarks

A video watermark shall be considered verified by an audio watermark when either the server code in the two watermarks is the same or the server code in the two watermarks is not the same but the video watermark identifies components in the same service as defined by the channel element (as defined in clause 7.1.2) of the AIT retrieved from a URL formed using watermark payload data detected in the audio watermark.

6.3.3 Monitoring for watermarks

Terminals that support application discovery based on ATSC3 watermarks as defined in the present document shall monitor for audio watermarks [12] and optionally video watermarks [13] when displaying content received over HDMI and optionally SCART, DVB transmission, and other external inputs for which application signalling may be unsupported or unavailable. For audio watermarks, terminals shall support detection of Audio Watermark Segments conveying the VP1 payload and for video watermarks, terminals shall support detection of Video Watermark Segments conveying VP1 Messages and HbbTV® Dynamic Event Messages.

Terminals not supporting the combination of application discovery over broadband as defined in the present document and targeted advertising as defined in TS 103 736-1 [15] should continue to monitor for watermarks on the external input when the HbbTV® application, launched as a result of detecting a previous watermark, is playing content from broadband and not displaying content received over HDMI. Terminals supporting the combination of application discovery over broadband as defined in the present document and as defined in TS 103 736-1 [15] shall continue to monitor for watermarks on the external input when the HbbTV® application, launched as a result of detecting a previous watermark, is playing content from broadband not displaying content received over HDMI.

Clauses 6.3.1 and D.2 refer to audio mute being performed either upstream of the HbbTV terminal on the STB or on the HbbTV terminal itself. When audio mute is applied on the terminal (i.e. not on the STB) then the terminal shall continue monitoring for watermarks in the received audio.

Clause 10.2.2 of the present document defines how applications can query:

1) which technologies the terminal supports for the control of application lifecycle;

2) if the terminal supports playing broadband-delivered video overlaid on video received over HDMI;

3) if watermark monitoring stops when broadband-delivered video is presented.

NOTE: Terminals may not have the ability to extract watermarks from HDMI content at the same time as playing content over broadband.

6.4 Application lifecycle when controlled by watermarks

6.4.1 Introduction

The application discovery process may be undertaken by an HbbTV® Terminal to discover and manage the lifecycle of HbbTV® applications. In accordance with the state machine specified in clause 6.3.1, application discovery is started, followed, and ends based on the detection and loss of watermarks. Clause 6.4.2 specifies terminal behaviours associated with starting and following the application discovery process. Clause 6.4.3 specifies terminal behaviours associated with ending it.

During application discovery, data obtained from watermarks is used to retrieve AITs for use in supporting application discovery. The data conveyed in audio and video watermarks may be different, so when both are present the terminal shall perform the application discovery process using data from the type of watermark as indicated in clause 6.3.1.

AITs delivered via application discovery using watermarks shall include extensions as specified in clause 7.1.2. These elements provide the Terminal with information about the current service that would typically be provided as broadcast signalling such as service identification information, timing information, and stream events but which is not conveyed over an HDMI interface.

6.4.2 Managing the HbbTV® application lifecycle

6.4.2.1 Introduction

Clause 6.3.1 specifies watermark-related events that cause the application discovery process to be started. To do this, the terminal shall acquire and process an AIT as specified in clause 6.4.2.2.

After the application discovery process is started, and until loss of watermark occurs per clause 6.3.1, the terminal shall continue to follow the HbbTV® application discovery process as specified in the present clause so long as a valid AIT (see clause 6.4.2.3) is available to the terminal.

AITs may indicate the period of the watermark media timeline for which they are valid (see clause 6.4.2.3). AITs may indicate they are not valid after a certain time on the media timeline in order to trigger reloading the AIT as that time is reached during playback, to enable a broadcaster to modify AITs in accordance with their program schedule. AITs may also indicate they are valid indefinitely.

When the current playback position on the watermark media timeline (see clause 6.4.2.4) approaches the end of the validity period of a valid AIT (see clause 6.4.2.3), the terminal shall perform a "scheduled AIT update" where it acquires and processes another AIT as specified in clause 6.4.2.2. The time period during which the terminal shall retrieve the AIT is measured backwards from the end of the validity period. The duration of this time period shall either be defined by the scheduledQuerySpread (defined in table 12), if present, otherwise it shall be 150 s. The Terminal should randomly select a time within this time period to make the AIT retrieval request so as to spread transaction load on the AIT server from any individual terminal across the requested time period interval. The terminal should not leave this until the very end of the validity period but allow for network and processing latencies so as to ensure that an updated AIT is received by the Terminal prior to the end of the AIT validity period.

If, at any time while a broadcast-related application is running whose lifecycle is controlled by watermarks, the current playback position as determined by watermarks in accordance with clause 6.4.2.4 reaches a point on the watermark media timeline for which the AIT is not valid in accordance with clause 6.4.2.3 (e.g. by failure of an AIT update request to return a valid AIT update or a media timeline discontinuity), the terminal shall:

1) if no attempt to acquire and process an AIT in accordance with clause 6.4.2.2 is currently underway, initiate that process; and
2) if the process for “loss of watermark” as given in clause 6.4.3 is not currently underway, initiate that process.

Table 8 of clause 6.3.1 specifies watermark-related events that cause an updated AIT to be retrieved based on changes to the Query Flag in the watermark data. The terminal shall maintain a single Query Flag value shared across audio and verified video watermarks. When a VP1 payload is detected in either the audio or verified video watermark with a different Query Flag value from the shared value and when a corresponding action is indicated, the terminal shall acquire and process another AIT as specified in clause 6.4.2.2 unless another Query Flag change has been detected within the past 1,5 s on the watermark media timeline, in which case the later Query Flag change shall be ignored.

NOTE: Care is needed to prevent missing changes in the Query Flag if values change in successive 1,5 s intervals.

When querySpread defined in table 12 is present, this value shall specify the maximum duration that the terminal is recommended to delay submission of the AIT request after detection of the change to the Query Flag. The Terminal should randomly select a time, up to the querySpread value, to delay the issuance of the AIT retrieval request so as to spread transaction load on the AIT server across the requested time period. If no querySpread is present then the terminal shall acquire and process the new AIT as quickly as possible.

An example scenario with "State Change in the Query Flag" events is illustrated in figure 4. In this example, VP1 Cells start at times 0 s, 1,5 s, and 3 s in the audio watermark and VP1 Message Groups start at times 0,5 s, 2 s, and 3,5 s in the video watermark. The broadcaster decides to initiate a state change in the Query Flag at time 1,8 s. The new query flag value begins being transmitted in the video watermark at time 2,0 s, when the next VP1 Message Group begins. This message can be detected from a single transmitted video frame, so the receiver is able to recognize the query flag change and issue a request to retrieve an updated AIT with 1 video frame of latency at time 2,0333 s (assuming a video frame rate of 30 fps). At time 3 s, the terminal has received the complete VP1 Cell which began at time 1,5 s, prior to the broadcaster initiating the Query Flag change. This payload carries the prior Query Flag value, which is different from the changed value being carried in the video watermark. Because a Query Flag change was detected within the prior 1,5 s, this change is ignored. Subsequent detections of the audio and video watermarks do not indicate any further Query Flag changes.

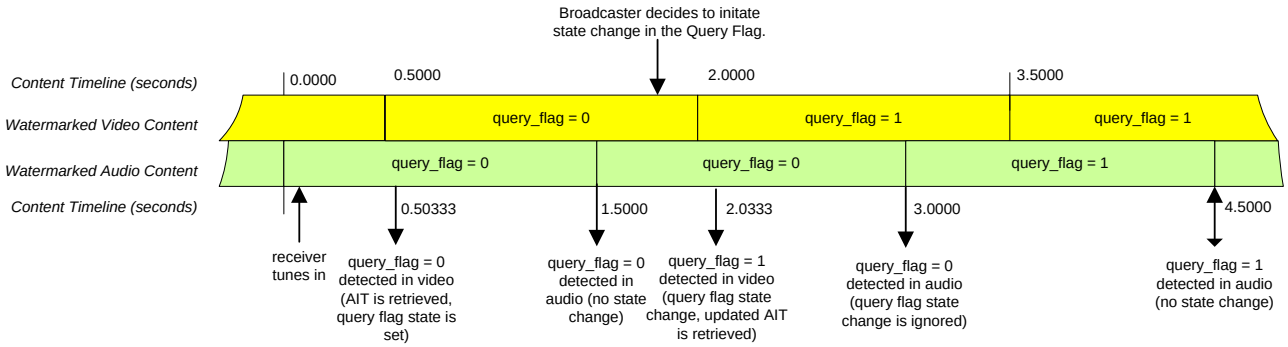

Figure 4: "State Change in the Query Flag" Example Scenario

If an attempt to retrieve an AIT in accordance with clause 6.4.2.2 fails to return an AIT (e.g. due to DNS failure, network timeout, or TLS certificate error), the terminal shall retry the same retrieval request repeatedly using an exponential backoff method where the initial delay is 5 s and succeeding delays are doubled until either:

a) an AIT is obtained via retry;

b) a scheduled AIT update is initiated in accordance with the present clause;

c) the process for loss of watermark or Query Flag change is started as indicated by operation of the state machine described in clause 6.3.1.

If an attempt to retrieve an AIT in accordance with clause 6.4.2.2 returns an invalid AIT, the terminal may again perform the process described in clause 6.4.2.2 but shall not do so with a frequency of greater than once per 5 s until either:

a) a valid AIT is retrieved;

b) a scheduled AIT update is initiated in accordance with the present clause; or

c) the process for loss of watermark or Query Flag change is started as indicated by operation of the state machine described in clause 6.3.1.

The following limitations shall apply to a broadcast-related application whose lifecycle is controlled by watermarks if the playback rate (as defined in clause 6.4.2.4.4) is negative.

- Acquiring and processing AITs (as defined in clause 6.4.2.2) shall be disabled.
- Stream events (as defined in clause 9.3) shall not be sent to applications.

When the playback rate is positive, the above limitations may also apply when content is received at the terminal at rates other than 1.0 (see the minimum requirements on determining playback rates in clause 6.4.2.4.4).

6.4.2.2 Acquire and Process an AIT

The terminal shall acquire and process AITs by performing the following steps, in sequence:

1) If any of the following conditions a) to d) apply:

   a) AIT acquisition and processing was initiated as a result of a Query Flag change as specified in table 8 of clause 6.3.1;

   b) the terminal is performing a scheduled AIT update per clause 6.4.2.1;

   c) no AIT previously obtained using watermark data is present on the Terminal;

   d) an AIT previously obtained using watermark data is cached in the Terminal but it is not valid for the presently detected watermark data in accordance with clause 6.4.2.3;

   and the current playback rate has been calculated by the terminal and is greater than zero then the terminal shall perform the steps i) to vi) in the order listed:

i) construct an HbbTV® DNS FQDN according to clause 5.4.2;

ii) if the results of a previous successful DNS resolution of the Authoritative FQDN from the HbbTV® DNS FQDN is cached by the terminal, the terminal shall use the cached result as the Authoritative FQDN; otherwise, if no negative response is cached by the terminal, the terminal shall perform a DNS request to obtain the Authoritative FQDN according to clause 5.5 and cache the server_field value in the detected watermark and the DNS resolution result according to clause 5.2;

iii) use the Authoritative FQDN to retrieve the AIT for the service according to clause 5.6.2;

iv) initialize the watermark media timeline as defined in clause 6.4.2.4;

v) confirm that the retrieved AIT is currently valid as defined in clause 6.4.2.3;

vi) confirm that the value of the queryFlag element, if present, of the current component object of the AIT as specified in clause 6.4.2.4.2 associated with the media type (audio or video) from which the watermark used to form the AIT retrieval request was detected matches the value in the corresponding field of the watermark used to form the AIT retrieval request. If the queryFlag element is present in the AIT but does not match the value detected from the watermark, the AIT shall be deemed invalid and the terminal shall not perform any AIT retrieval action as a result of any Query Flag state change event for 5 minutes or until loss of watermark occurs, if sooner than 5 minutes.

else, the AIT that was previously obtained using watermark data shall be employed.

2) If the terminal now has a valid AIT for the service to employ, then:

a) If the service identified by the AIT is the same as the service identified by the most recently active AIT, then the terminal shall apply clause 6.2.2.3 of TS 102 796 [1] using the AIT, however the step "app continues to run" in that clause shall be: "app continues to run and is unhidden, if it was hidden according to 6.4.3 of TS 103 464".

b) Otherwise, it shall apply clause 6.2.2.2 of TS 102 796 [1] using that AIT, however the step "app continues to run" in that clause shall be: "app continues to run and is unhidden, if it was hidden according to 6.4.3 of TS 103 464".

c) If a running application is unhidden and has a video/broadcast object that is in the Connecting state then that object shall transition to the Presenting state.

3) If the terminal has performed the preceding steps and does not have a valid AIT corresponding to a service identified by watermark data, it shall be treated the same as a selection of a service with no AIT as defined TS 102 796 [1].

6.4.2.3 AIT Validity

When application discovery is performed using watermarks, an AIT shall be considered valid for the presently detected watermark when the following conditions are all met:

1) The server code in the presently detected watermark is present in either a videoComponent or an audioComponent (as appropriate for the type of media carrying the watermark) as defined in table 12, in order to verify that the component from which the watermark is detected is a component of the service described by the AIT.

2) The current playback position in the service as determined by the presently detected watermark (see clause 6.4.2.4) is within the validity period of the AIT defined by the attribute values of validFrom and validUntil of the AIT in table 11.

6.4.2.4 Watermark media timeline

6.4.2.4.1 Introduction

The watermark media timeline is a stable reference frame for associating media time to watermarked content (i.e. a timebase timeline per TS 103 286-2 [i.1], similar to TEMI (see ISO/IEC 13818-1:2018 [i.3]) that enables terminals and

HbbTV® applications to precisely synchronize elements of the media presentation, regardless of distribution path latency when application discovery using watermarking is employed.

The watermark media timeline is established by the temporal location of the watermarks in audio and video components of a service, the interval_field values carried in the watermark payload, and the @audioComponent and @videoComponent elements conveyed in XML AIT extensions. An example is shown in figure 5.

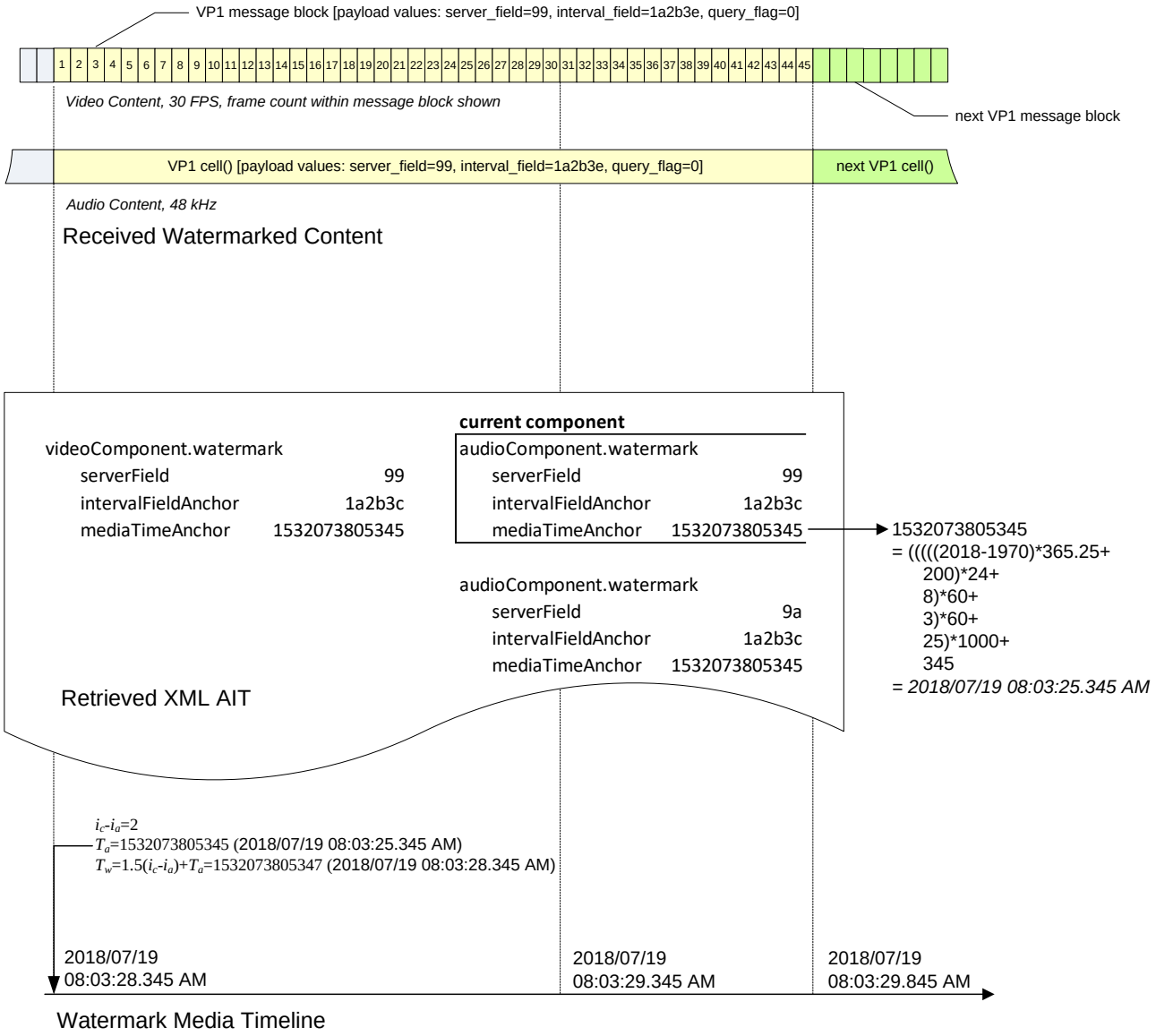

Figure 5: Watermark media timeline example

NOTE: The above figure assumes broadcasters encode time since 1/1/1970 in the watermark media timeline (using intervalFieldAnchor and mediaTimeAnchor as defined in clause 6.4.2.4.2). It is optional for broadcasters to do this and other possibilities exist, e.g. resetting watermark media time at a specific time in the middle of the night.

The watermark media timeline is used in the application discovery process by the terminal to associate a media time with the current playback position in the content for purposes of:

1) determining AIT validity as specified in clause 6.4.2.3;

2) performing scheduled AIT updates as specified in clause 6.4.2.1; and

3) supporting use of the MediaSynchronizer and stream event APIs when application discovery using watermarking is employed (see clauses 9.2 and 9.3).

6.4.2.4.2 Initializing the watermark media timeline

The terminal shall select a current audio component from among those audioComponent elements present in the AIT with watermark@serverField value equal to the server_field value present in the most recently detected audio watermark (the "candidate audio components") as follows:

1) If there are any candidate audio components with watermark@intervalFieldAnchor value less than or equal to the detected interval_field, then the current audio component shall be the candidate audio component whose watermark@intervalFieldAnchor value is nearest to, but not greater than, the interval_field value in the most recently detected audio watermark (if more than one candidate audio component meets this selection criterion, then the first such component listed in the AIT shall be selected).

2) Otherwise, the current audio component shall be the candidate audio component whose watermark@intervalFieldAnchor value is nearest to the interval_field value in the most recently detected audio watermark (if more than one candidate audio component meets this selection criterion, then the first such component listed in the AIT shall be selected).

3) If no such component is present in the AIT, then no current audio component shall be selected.

If the watermark state machine (defined in clause 6.3.1) is in either the Audio and verified video watermark detected state or the Verified video watermark detected only state, then the terminal shall select a current video component from among those videoComponent elements in the AIT with watermark@serverField value equal to the server_field value present in the most recently detected video watermark (the "candidate video components") as follows:

1) If there are any candidate video components with watermark@intervalFieldAnchor less than or equal to the detected interval_field, then the current video component shall be the candidate video component whose watermark@intervalFieldAnchor value is nearest to, but not greater than, the interval_field value in the most recently detected video watermark (if more than one candidate video component meets this selection criterion, then the first such component listed in the AIT shall be selected).

2) Otherwise, the current video component shall be the candidate video component whose watermark@intervalFieldAnchor value is nearest to the interval_field value in the most recently detected video watermark (if more than one candidate video component meets this selection criterion, then the first such component listed in the AIT shall be selected).

3) If no such component is present in the AIT, then no current video component shall be selected.

If the watermark state machine (defined in clause 6.2.1) is in one of the following states then the watermark media timeline shall be initialized from the audio watermark detected:

- Audio watermark detected only.
- Audio and unverified video watermarks detected.
- Audio and verified video watermarks detected.

If the watermark state machine (defined in clause 6.2.1) is in one of the following states then the watermark media timeline shall be initialized from the video watermark detected:

- Verified video watermark detected only.

The following process shall be followed when the terminal is required to initialize the media timeline from an audio watermark and an AIT:

1) If no current audio component is selected, the watermark media timeline is considered to have not been initialized.

2) If a current audio component is selected, the terminal shall initialize the watermark media timeline such that the media time of the first audio sample of the first symbol of the most recently detected audio watermark cell has the media time (in milliseconds):

$$((i_w - i_a) \times 1\,500) + T_a,$$

where:

i_w = the value of the interval_field in the audio watermark

i_a = the value of watermark@intervalFieldAnchor in the current audio component

T_a = the value of watermark@mediaTimeAnchor in the current audio component

The following process shall be followed when the terminal is required to initialize the media timeline from a video watermark and an AIT.

1) If no current video component is selected, the watermark media timeline is considered to have not been initialized.

2) If a current video component is selected, the terminal shall initialize the watermark media timeline such that;

   a. if the video watermark payload conveys a VP1 message with a time_offset field, the media time of the video frame in which the most recently detected payload is carried is (in milliseconds):

      $((i_w – i_a) * 1500) + T_a + (1000 / 30) * o_w$,

      where

      i_w = the value of the interval_field in the video watermark

      i_a = the value of watermark@intervalFieldAnchor in the current video component

      T_a = the value of watermark@mediaTimeAnchor in the current video component

      o_w = the value of the time_offset in the video watermark;

   b. otherwise, the media time of the video frame carrying the first video watermark payload of the most recently detected video watermark message group has the media time (in milliseconds):

      $((i_w – i_a) \times 1\ 500) + T_a$,

where:

i_w = the value of the interval_field in the video watermark

i_a = the value of watermark@intervalFieldAnchor in the current video component

T_a = the value of watermark@mediaTimeAnchor in the current video component

6.4.2.4.3 Determining the Media Time of watermarked content

After the watermark media timeline is initialized, it provides a mapping for associating a media time with points in time in the watermarked content.

During an audio watermark segment (either with or without the detection of video watermarks), the media time associated with each audio sample in the watermark segment shall be determined based on its offset in the media stream relative to the audio sample at which the watermark media timeline was initialized. That is, the media time of each audio sample shall be offset from that of the adjacent audio samples by the nominal audio sampling interval times the playback rate.

While the terminal is in the watermark state "Verified Video Watermark Detected Only" (as specified in clause 6.3.1), the media time associated with each video frame in the video watermark segment shall be determined based on its offset in the media stream relative to the video frame at which the watermark media timeline was initialized. That is, the media time of each video frame shall be offset from that of the adjacent video frames by a value equal to the nominal video frame interval times the playback rate.

NOTE: Media time calculations should be made with sufficient precision to avoid drift due to accumulated rounding error.

The application lifecycle requirements in clause 6.4.2.1 relating to the playback position changing shall be followed when applicable (e.g. the AIT becoming invalid or being close to becoming invalid).

6.4.2.4.4 Maintaining synchronization to the Watermark Media Timeline

During an audio watermark segment (either with or without the detection of video watermarks), after the watermark media timeline for an audio watermark segment has been initialized in accordance with clause 6.4.2.4.2, upon detection of each subsequent watermark payload in an audio watermark segment, terminals shall calculate:

1) the watermark media time according to clause 6.4.2.4.3; and

2) the watermark media time that would apply if the watermark media timeline was initialized according to clause 6.4.2.4.2 using the newly received watermark payload; and

3) the playback rate as a multiplicative factor indicating the amount of temporal scaling observed in the detected audio watermark relative to its normative specification (i.e. 1.0 is the normative playback rate). An example method for determining the playback rate from the audio watermark is to divide the expected time difference, according to the normative specification, between the two most recently detected audio watermark payloads by their observed time difference in the monitored input stream (i.e. if VP1 payloads containing successive interval_field values are detected separated in the monitored stream by 1,485149 s of audio samples, the playback rate can be calculated as 1,5 s / 1,485149 s = 1,01). Terminals shall support determining playback rates from content received at the terminal at playback rate 1.0. Calculation of other playback rates is implementation specific.

If the two media times differ by greater than one-half the video frame interval (i.e. one-half divided by the video frame rate), the terminal shall re-initialize the watermark media timeline as specified in clause 6.4.2.4.2.

While the terminal is in the watermark state "Verified Video Watermark Detected Only" (as specified in clause 6.3.1), after the watermark media timeline has been initialized in accordance with clause 6.4.2.4.2, upon detection of each subsequent VP1 message with a time_offset field or video watermark message group in a video watermark segment, terminals shall re-initialize the watermark media timeline as specified in clause 6.4.2.4.2. Terminals shall calculate the playback rate expressed as a multiplicative factor indicating the amount of temporal scaling observed in the video watermark relative to its normative specification (i.e. 1.0 is the normative playback rate). An example method for estimating the playback rate of the video watermark is to divide the expected time difference, according to the normative specification, between the first detected instance of the two unique VP1 Messages by their observed time difference in the monitored input stream (i.e. if VP1 messages containing the same interval_field values and time_offset values 1 and 19 – indicating a normative timeline separation of 0,6 s - are first detected in frames separated in the monitored stream by 0,4 s of video frames, the playback rate can be calculated as 0,6 s / 0,4 s = 1,5). Terminals shall support determining playback rates from content received at the terminal at playback rates of 0.0 and 1.0. Calculation of other playback rates is implementation specific.

NOTE 1: Fast forward / fast rewind algorithms in STBs may behave very differently and the present document is intentionally silent about calculation of playback rate under these circumstances.

NOTE 2: Jitter may be present in the watermark media timeline when the video watermark is carried in 25 or 50 Hz frame rate video and time_offset values are conveyed with 1/30 s resolution. This jitter can impact playback rate estimates if the time different measurements are made over short time intervals. Terminal implementers can limit the impact of this jitter on playback rate estimates to within +/-0.05 by using video watermark detections separated by at least 0,5 s in the monitored stream. Broadcasters are advised that playback rates reported by the terminal will include some estimation error.

The application lifecycle requirements in clause 6.4.2.1 relating to the playback position changing shall be followed when applicable (e.g. the AIT becoming invalid or being close to becoming invalid).

6.4.3 Loss of watermark

Clause 6.3.1 specifies the conditions under which a loss of watermark detection condition is indicated. When this happens, the following shall apply:

- If a broadcast-related HbbTV® application is running that the user has not activated (see clause 10.2.2.1 of TS 102 796 [1]), that application shall be stopped as defined in clause 6.2.2.2 of TS 102 796 [1] for a broadcast service where no HbbTV® applications are signalled.

- If a broadcast-related HbbTV® application is running that the user has activated (see clause 10.2.2.1 of TS 102 796 [1]) then terminals should hide the application and allow the user to see the whole area of the content

delivered by HDMI within 2 s. If the application has a video/broadcast object that is in the Presenting state then that object shall transition to the Connecting state.

- Terminals should stop applications that are hidden pursuant to this clause (and have not been subsequently killed or unhidden according to clause 6.4.2) after an implementation-dependent period, not less than 2 minutes.

NOTE: Clause 6.4.2.2 defines the conditions when an application that is running but hidden according to this clause is required to stop being hidden.

- If a running application has a switch scheduled and the application is not stopped then the following shall apply;
  - Switches from broadcast to an advert (or other content) shall be aborted and the promise returned by the call to `switchMediaPresentation` shall be rejected with an `InvalidStateError`.
  - Switches from an advert (or other content) to broadcast shall be executed immediately and the promise returned by the call to `switchMediaPresentation` shall be resolved with `undefined`.

6.4.4 Transitioning between HbbTV® Application types

Terminals shall support one broadcast-related HbbTV® application starting another broadcast-related HbbTV® application based on "dvb:" URLs and use the AIT retrieved using watermark data for the current channel to resolve the URL.

If a broadcast-related HbbTV® application starts a broadcast-independent application then the lifecycle of that second application shall not be controlled by the watermark.

If a broadcast-related HbbTV® application calls the setChannel() method on the video/broadcast object with a value of null for its channel argument then the lifecycle of that application shall stop being controlled by the watermark and transition to become broadcast independent.

NOTE: Care is needed in both the above circumstances as any changes made on the STB by the end-user will be invisible. The design of the application needs to clearly communicate to the end-user that they are moving from one context to a different context and either that the output of the STB is no longer visible or that the STB remote is no longer active.

If an HbbTV® application that was originally launched as broadcast-related following watermark detection transitions to broadcast-independent and then some time later attempts to transition back to broadcast related then the terminal shall present the video and audio from the HDMI input last used with HbbTV® and attempt to detect watermarks. If a watermark is detected then the terminal shall decide whether the application is killed by applying the conditions in clause 6.2.2.6.1 of TS 102 796 [1] including obtaining the AIT. Broadcast-independent applications not originally launched as broadcast-related following watermark detection shall be stopped if they attempt to transition to broadcast-related.

If a broadcast-independent HbbTV® application attempts to transition to broadcast-related as defined in clause 6.2.2.6.1 of TS 102 796 [1], it shall be stopped.

If the content with the video watermark is encoded at very low bitrates (e.g. MPEG-2 at a fixed bitrate of 1 Mbit/s), it may not be possible to reliably extract the video watermark. It is not possible to provide an exact threshold, though reliable operation using MPEG-2 at 2,5 Mbit/s has been demonstrated. Reliability can depend on many factors, such as the codec type, the codec model, how that codec allocates its bits across the frame, the bitrate, whether a statistical multiplex is used, the nature of the content, etc. Broadcasters can adjust their use of the technology, including the watermark embedding level, the redundancy of dynamic event message transmission, and application behaviour accordingly. Applications can monitor video watermark reliability using the watermark state tracking capability specified in clause 8.1.

7 Formats and protocols

7.1 Signalling of applications

7.1.1 XML AIT for Broadcast-related broadband application discovery

The AIT for broadcast-related applications discovered over broadband shall be encoded in XML similarly to the way defined in TS 102 796 [1] for broadcast-independent applications. The XML file shall contain an ApplicationDiscovery record containing one or more `<application>` elements.

The semantics of the fields and elements in the XML AIT file shall be as defined in table 10. All entries which are identical to the contents as defined in table 7 "Contents of XML AIT for Broadcast-independent applications" of TS 102 796 [1] are marked as "Same as for broadcast-independent applications".

Table 10: Contents of XML AIT for Discovered Broadcast-related Applications

Field or element	Requirement on XML AIT file	Requirement on terminal
appName	Same as for broadcast-independent applications.	Same as for broadcast-independent applications.
applicationIdentifier	Same as for broadcast-independent applications.	Same as for broadcast-independent applications.
applicationDescriptor/type/OtherApp	Same as for broadcast-independent applications.	Same as for broadcast-independent applications.
applicationDescriptor/controlCode	Mandatory.	Mandatory.
applicationDescriptor/visibility	Same as for broadcast-independent applications.	Same as for broadcast-independent applications.
applicationDescriptor/serviceBound	Mandatory.	Mandatory.
applicationDescriptor/priority	Same as for broadcast-independent applications.	Same as for broadcast-independent applications.
applicationDescriptor/version	Same as for broadcast-independent applications.	Same as for broadcast-independent applications.
applicationDescriptor/mhpVersion	Shall be the same values as defined for the MPEG-2 encoding of the AIT in the row of table 5 "Supported application signalling features" from TS 102 796 [1] that corresponds to clause 5.2.5 of TS 102 809 [6].	As defined in table 7 "Contents of XML AIT for Broadcast-independent applications" of TS 102 796 [1].
applicationDescriptor/icon	Same as for broadcast-independent applications.	Same as for broadcast-independent applications.
applicationDescriptor/storageCapabilities	Same as for broadcast-independent applications.	Same as for broadcast-independent applications.
applicationTransport/	Same as for broadcast-independent applications.	Same as for broadcast-independent applications.
applicationLocation/	Same as for broadcast-independent applications.	Same as for broadcast-independent applications.
applicationBoundary/	Same as for broadcast-independent applications.	Same as for broadcast-independent applications.
applicationSpecificDescriptor	Same as for broadcast-independent applications.	Same as for broadcast-independent applications.
applicationUsageDescriptor	Same as for broadcast-independent applications.	Same as for broadcast-independent applications.

Clause 7.2.3.2 of TS 102 796 (V1.4.1) [2] defines how to signal parental rating information in the XML AIT - something not included in some earlier versions. When the solution defined in the present document is used in combination with TS 102 796 (V1.4.1) [2], or a more recent version of TS 102 796 [1], if the XML AIT for a Discovered broadcast-related application includes a parental rating then that rating shall be acted upon as defined in clause 6.2.2.10 of TS 102 796 (V1.4.1) [2]. Parental rating for Discovered broadcast-related applications is not supported when the solution defined in the present document is used with earlier versions of TS 102 796 [1].

Optionally XML DSIG [8] can be used to authenticate XML AITs and hence to enable terminals to reject XML AITs if (for example) DNS responses are being tampered with. Terminals not supporting this option shall silently ignore any `Signature` element that appears within the `ServiceDiscovery` element and process the XML AIT as if that information were not present.

NOTE: The present document does not define a trust hierarchy for authenticating XML AITs.

7.1.2 XML AIT Extensions

When used with watermarking, the XML AIT format described in clause 5.4 of TS 102 809 [6] and extended in clause 7.2.3.2 of TS 102 796 [1] is further extended as defined in the following schema and in tables 11 and 12. The normative definition of this schema is found in the electronic attachments - see annex B of the present document.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:hbbwm="urn:hbbtv:watermark:2018" xmlns:ait="urn:dvb:mhp:2009"
           xmlns:dvb="urn:dvb:metadata:iptv:sdns:2008-1"
           xmlns:oipf="urn:oipf:iptv:ContentAccessDownloadDescriptor:2008-1"
           xmlns:hbbtv="urn:hbbtv:application_descriptor:2014"
           targetNamespace="urn:hbbtv:watermark:2018"
           elementFormDefault="qualified" attributeFormDefault="unqualified" >
    <xs:import namespace="urn:dvb:mhp:2009" schemaLocation="oipf/imports/mis_xmlait.xsd"/>
    <xs:import namespace="urn:dvb:metadata:iptv:sdns:2008-1"
               schemaLocation="oipf/imports/sdns_v1.4r13.xsd"/>
<xs:import namespace="urn:oipf:iptv:ContentAccessDownloadDescriptor:2008-1"
               schemaLocation= "oipf/iptv-ContentAccessDownloadDescriptor.xsd"/>
    <xs:import namespace="urn:hbbtv:application_descriptor:2014"
               schemaLocation="hbbtv_application_descriptor.xsd"/>
    <xs:complexType name="WatermarkExtensions">
        <xs:complexContent>
            <xs:extension base="ait:ApplicationOfferingType">
                <xs:sequence>
                   <xs:element name="channel" type="hbbwm:ChannelType"/>
                   <xs:element name="validFrom" type="xs:unsignedLong" minOccurs="0"/>
                   <xs:element name="validUntil" type="xs:unsignedLong" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ChannelType">
        <xs:sequence>
            <xs:element name="channelType" type="hbbwm:ChannelTypeType"/>
            <xs:element name="idType" type="hbbwm:ChannelIDType"/>
            <xs:element name="nid" type="xs:unsignedShort"/>
            <xs:element name="onid" type="dvb:OrigNetId"/>
            <xs:element name="tsid" type="dvb:TSId"/>
            <xs:element name="sid" type="dvb:ServiceId"/>
            <xs:element name="name" type="dvb:Service"/>
            <xs:element name="majorChannel" type="xs:unsignedShort"/>
            <xs:element name="videoComponent" type="hbbwm:AVVideoComponentType" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="audioComponent" type="hbbwm:AVAudioComponentType"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:simpleType name="ChannelTypeType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="TYPE_TV"/>
            <xs:enumeration value="TYPE_RADIO"/>
            <xs:enumeration value="TYPE_OTHER"/>
            <xs:enumeration value="TYPE_HBBTV_DATA"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="ChannelIDType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="ID_ANALOG"/>
            <xs:enumeration value="ID_DVB_C"/>
            <xs:enumeration value="ID_DVB_S"/>
            <xs:enumeration value="ID_DVB_T"/>
            <xs:enumeration value="ID_DVB_C2"/>
            <xs:enumeration value="ID_DVB_S2"/>
            <xs:enumeration value="ID_DVB_T2"/>
            <xs:enumeration value="ID_IPTV_SDS"/>
            <xs:enumeration value="ID_IPTV_URI"/>
        </xs:restriction>
```

```
    </xs:simpleType>
    <xs:complexType name="AVComponentType">
        <xs:sequence>
            <xs:element name="componentTag" type="xs:unsignedByte"/>
            <xs:element name="pid" type="dvb:Hexadecimal16bit"/>
            <xs:element name="type" type="hbbwm:ComponentTypeType"/>
            <xs:element name="encoding" type="xs:string"/>
            <xs:element name="encrypted" type="xs:boolean"/>
            <xs:element name="watermark" type="hbbwm:WatermarkComponentType"/>
        </xs:sequence>
    </xs:complexType>
    <xs:simpleType name="ComponentTypeType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="COMPONENT_TYPE_VIDEO"/>
            <xs:enumeration value="COMPONENT_TYPE_AUDIO"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="AVAudioComponentType">
        <xs:complexContent>
            <xs:extension base="hbbwm:AVComponentType">
                <xs:sequence>
                    <xs:element name="language" type="dvb:ISO639-2"/>
                    <xs:element name="audioDescription" type="xs:boolean"/>
                    <xs:element name="audioChannels" type="xs:short"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="AVVideoComponentType">
        <xs:complexContent>
            <xs:extension base="hbbwm:AVComponentType">
                 <xs:attribute name="aspectRatio" type="xs:decimal"/>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="WatermarkComponentType">
    </xs:complexType>
    <xs:complexType name="ATSC3WatermarkComponentType">
        <xs:complexContent>
            <xs:extension base="hbbwm:WatermarkComponentType">
                <xs:sequence>
                    <xs:element name="serverField" type="dvb:Hexadecimal32bit"/>
                    <xs:element name="intervalFieldAnchor" type="dvb:Hexadecimal32bit"/>
                    <xs:element name="mediaTimeAnchor" type="xs:unsignedLong"/>
                    <xs:element name="querySpread" type="xs:integer" minOccurs="0"/>
                    <xs:element name="scheduledQuerySpread" type="xs:integer" minOccurs="0"/>
                    <xs:element name="queryFlag" type="xs:boolean" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
</xs:schema>
```

An example of an XML AIT using this schema (informative). This example can also be found in the electronic attachments - see annex B of the present document.

```
<?xml version="1.0" encoding="UTF-8"?>
<ait:ServiceDiscovery
    xmlns:ait="urn:dvb:mhp:2009"
    xmlns:hbb="urn:hbbtv:application_descriptor:2014"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:hbbwm="urn:hbbtv:watermark:2018">
    <ait:ApplicationDiscovery xsi:type="hbbwm:WatermarkExtensions" DomainName="example.com">
        <ait:ApplicationList>
            <ait:Application>
                <ait:appName Language="eng">Whizzo Play Along Quiz</ait:appName>
                    <ait:applicationIdentifier>
                    <ait:orgId>123</ait:orgId>
                    <ait:appId>456</ait:appId>
                </ait:applicationIdentifier>
                <ait:applicationDescriptor xsi:type="hbb:HbbTVApplicationDescriptor">
                    <ait:type>
                        <ait:OtherApp>application/vnd.hbbtv.xhtml+xml</ait:OtherApp>
                    </ait:type>
                    <ait:controlCode>AUTOSTART</ait:controlCode>
                    <ait:visibility>VISIBLE_ALL</ait:visibility>
                    <ait:serviceBound>false</ait:serviceBound>
```

```
                    <ait:priority>1</ait:priority>
                    <ait:version>01</ait:version>
                    <ait:mhpVersion>
                        <ait:profile>0</ait:profile>
                        <ait:versionMajor>1</ait:versionMajor>
                        <ait:versionMinor>4</ait:versionMinor>
                        <ait:versionMicro>1</ait:versionMicro>
                    </ait:mhpVersion>
                    <hbb:ParentalRating Scheme="dvb-si" Region="GB">8</hbb:ParentalRating>
                </ait:applicationDescriptor>
                <ait:applicationTransport xsi:type="ait:HTTPTransportType">
                    <ait:URLBase>https://www.example.com/</ait:URLBase>
                </ait:applicationTransport>
                <ait:applicationLocation>whizzo-app.html?a=1</ait:applicationLocation>
            </ait:Application>
        </ait:ApplicationList>
            <hbbwm:channel>
                <hbbwm:channelType>TYPE_TV</hbbwm:channelType>
                <hbbwm:idType>ID_DVB_S2</hbbwm:idType>
                <hbbwm:nid>123</hbbwm:nid>
                <hbbwm:onid>124</hbbwm:onid>
                <hbbwm:tsid>3</hbbwm:tsid>
                <hbbwm:sid>99</hbbwm:sid>
                <hbbwm:name>"Example TV Channel"</hbbwm:name>
                <hbbwm:majorChannel>7</hbbwm:majorChannel>
                <hbbwm:videoComponent aspectRatio="1.78">
                      <hbbwm:componentTag>1</hbbwm:componentTag>
                      <hbbwm:pid>72</hbbwm:pid>
                      <hbbwm:type>COMPONENT_TYPE_VIDEO</hbbwm:type>
                      <hbbwm:encoding></hbbwm:encoding>
                      <hbbwm:encrypted>true</hbbwm:encrypted>
                      <hbbwm:watermark xsi:type="hbbwm:ATSC3WatermarkComponentType">
                            <hbbwm:serverField>9999ffff</hbbwm:serverField>
                            <hbbwm:intervalFieldAnchor>1a2b3c9a</hbbwm:intervalFieldAnchor>
                            <hbbwm:mediaTimeAnchor>1532073805345</hbbwm:mediaTimeAnchor>
                      </hbbwm:watermark>
                </hbbwm:videoComponent>
                <hbbwm:audioComponent>
                      <hbbwm:componentTag>10</hbbwm:componentTag>
                      <hbbwm:pid>98</hbbwm:pid>
                      <hbbwm:type>COMPONENT_TYPE_AUDIO</hbbwm:type>
                      <hbbwm:encoding></hbbwm:encoding>
                      <hbbwm:encrypted>true</hbbwm:encrypted>
                      <hbbwm:watermark xsi:type="hbbwm:ATSC3WatermarkComponentType">
                            <hbbwm:serverField>9999ffff</hbbwm:serverField>
                            <hbbwm:intervalFieldAnchor>1a2b3c9a</hbbwm:intervalFieldAnchor>
                            <hbbwm:mediaTimeAnchor>1532073805345</hbbwm:mediaTimeAnchor>
                            <hbbwm:queryFlag>true</hbbwm:queryFlag>
                      </hbbwm:watermark>
                      <hbbwm:language>eng</hbbwm:language>
                      <hbbwm:audioDescription>false</hbbwm:audioDescription>
                      <hbbwm:audioChannels>5</hbbwm:audioChannels>
                </hbbwm:audioComponent>
                <hbbwm:audioComponent>
                      <hbbwm:componentTag>11</hbbwm:componentTag>
                      <hbbwm:pid>99</hbbwm:pid>
                      <hbbwm:type>COMPONENT_TYPE_AUDIO</hbbwm:type>
                      <hbbwm:encoding></hbbwm:encoding>
                      <hbbwm:encrypted>true</hbbwm:encrypted>
                      <hbbwm:watermark xsi:type="hbbwm:ATSC3WatermarkComponentType">
                            <hbbwm:serverField>1000000A</hbbwm:serverField>
                            <hbbwm:intervalFieldAnchor>1a2b3c84</hbbwm:intervalFieldAnchor>
                            <hbbwm:mediaTimeAnchor>1532073805345</hbbwm:mediaTimeAnchor>
                      </hbbwm:watermark>
                      <hbbwm:language>deu</hbbwm:language>
                      <hbbwm:audioDescription>false</hbbwm:audioDescription>
                      <hbbwm:audioChannels>5</hbbwm:audioChannels>
                </hbbwm:audioComponent>
            </hbbwm:channel>
    </ait:ApplicationDiscovery>
</ait:ServiceDiscovery>
```

The extended element definitions shown in table 11 shall apply to XML AITs retrieved using information obtained from watermarking.

Table 11: AIT extensions for watermarking

Parent	Element or Attribute Name	Optional (Informative)	Max elements (Informative)	Format
WatermarkExtensions	validFrom	Yes	1	Earliest time (milliseconds) on the watermark media timeline at which the AIT is valid. When absent, there is no earliest time at which the AIT is valid (i.e. the AIT is valid at any time prior to the latest validity time).
	validUntil	Yes	1	Latest time (milliseconds) on the watermark media timeline at which the AIT is valid. When absent, there is no latest time at which the AIT is valid (i.e. the AIT is valid at any time after the earliest validity time).
	channel	No	1	Container for the properties of a broadcast TV channel and its components
Channel	channelType	No	1	Shall have the same meaning as the property with the same name defined in the `Channel` class in clause 7.13.11 of the OIPF DAE specification [4].
	idType	No	1	
	nid	No	1	
	onid	No	1	
	tsid	No	1	
	sid	No	1	
	name	No	1	
	majorChannel	No	1	
	videoComponent	Yes	*	Container for the video components in the broadcast TV service.
	audioComponent	No	*	Container for the audio components in the broadcast TV service.
AVComponent	componentTag	No	1	Shall have the same meaning as the property with the same name defined in the `AVComponent` class in clause 7.16.5.2.1 of the OIPF DAE specification [4].
	pid	No	1	
	type	No	1	
	encoding	No	1	
	encrypted	No	1	
	watermark	No	1	Provides additional information about AVComponents that can be identified by a watermark. See Table 12.
AVAudioComponent	language	No	1	Shall have the same meaning as the property with the same name defined in the `AVAudioComponent` class in clause 7.16.5.4.1 of the OIPF DAE specification [4].
	audioDescription	No	1	
	audioChannels	No	1	
AVVideoComponent	aspectRatio	No	1	Shall have the same meaning as the property with the same name defined in the `AVVideoComponent` class in clause 7.16.5.3.1 of the OIPF DAE specification [4].

The extended element definitions shown in table 12 shall apply to XML AITs retrieved using information from watermarks.

Table 12: XML AIT Extensions for Application Discovery over Broadband with Watermarks

Parent	Element or Attribute Name	Optional	Format
ATSC3WatermarkComponentType	serverField	No	The VP1 Server Code (as defined in ATSC A/336 [9]) used in the watermark of the component matching this AVAudioComponent or AVVideoComponent.
	intervalFieldAnchor	No	The value of the VP1 Interval Code corresponding to the mediatimeAnchor.
	mediaTimeAnchor	No	The time on the watermark media timeline corresponding to a watermark payload whose Interval Code is contained in the IntervalFieldAnchor element. For an audio watermark component, this corresponds to the sampling instant of the first sample of the first symbol of the audio watermark Cell conveying that payload. For a video watermark component, this corresponds to the sampling instant of the first frame of the VP1 Message Group conveying that payload.
	querySpread	Yes	Time period in milliseconds on the wall-clock timeline during which terminal should request AIT retrieval when signalled by the Query Flag.
	scheduledQuerySpread	Yes	Time period in milliseconds on the watermark media timeline during which terminal should query AIT server when the end of the validity period approaches.
	queryFlag	Yes	The value of the VP1 (as defined in ATSC A/336 [9]) used in the watermark of the component matching this AVAudioComponent or AVVideoComponent.

7.2 Watermark formats

7.2.1 Introduction

The present document includes one audio watermark format and one video watermark format. The present document is structured to enable addition of other formats in the future.

The present document makes extensive use of the data structures defined by ATSC in A/336 [9] - particularly the VP1 payload but also the dynamic event message. These data structures are believed to be independent of the underlying watermark format and hence should be possible to re-use with other formats.

7.2.2 ATSC 3 Watermarks

The present document includes the ATSC 3 watermarks defined in A/334 [12] and A/335 [13].

If the content with the ATSC video watermark is encoded at very low bitrates (e.g. MPEG-2 at a fixed bitrate of 1 Mbit/s), it may not be possible to reliably extract the video watermark. It is not possible to provide an exact threshold, though reliable operation using MPEG-2 at 2,5 Mbit/s has been demonstrated. Reliability can depend on many factors, such as the codec type, the codec model, how that codec allocates its bits across the frame, the bitrate, whether a statistical multiplex is used, the nature of the content, etc. Broadcasters can adjust their use of the technology, including the watermark embedding level, the redundancy of dynamic event message transmission, and application behaviour accordingly. Applications can monitor video watermark reliability using the watermark state tracking capability specified in clause 8.1.

8 Browser application environment

8.1 Extensions to the application/oipfApplicationManager embedded object and the Application class

The following additional property shall be supported on the Application class.

`readonly String lifecycleControl`	
Description	Indicates which signalling mechanism is controlling the lifecycle of the application reading the property. Values shall be as follows. Broadcast-independent application. — "" (i.e. empty string) Broadcast AIT as defined in TS 102 796 [1]. — "ait-mpeg2" Discovered AIT based on DVB-SI as defined in clauses 5.3.1, 5.4.1 and 5.6.1 of the present document. — "xmlait-dvbsi" ATSC3 watermark as used in the present document. — "xmlait-atsc3"

The following additional property shall be supported on the application/oipfApplicationManager embedded object.

`readonly String watermarkState`	
Description	Returns the watermark state as defined in clause 6.3.1 of the present document. This shall be encoded as follows. Audio Watermark Detected Only. — wm-audio-only Audio and Unverified Video Watermarks Detected. — wm-audio-unverified-video No Watermark Detected. — wm-none Unverified Video Watermark Detected Only. — wm-unverified-video-only Verified Video Watermark Detected Only. — wm-verified-video-only Audio and Verified Video Watermarks Detected. — wm-audio-verified-video Watermark detector is not running (e.g. an application controlled by a broadcast AIT as defined in TS 102 796 [1] is running on a terminal supporting the present document but not supporting watermark detection from broadcast video). — wm-not-running Note that some of these values may only be returned transiently (e.g. when the loss of watermark process is being followed), may only be returned under very precise circumstances or may not be returned at all.

Applications shall be able to use the `addEventListener` method to register `EventListeners` for events of type "WatermarkStateChange" on the oipfApplicationManager object (as defined in the OIPF DAE specification [4]). When the watermark state changes (as defined in clause 6.3.1 of the present document), a `WatermarkStateChangedEvent` shall be sent to all `EventListener`s registered for events of that type.

```
interface WatermarkStateChangedEvent : Event {
    readonly attribute String      oldState;
    readonly attribute String      newState;
}
```

Properties	`oldState`	The watermark state before the state change. State names shall be encoded as defined for the `watermarkState` property.
	`newState`	The watermark state after the state change. State names shall be encoded as defined for the `watermarkState` property.

NOTE: This event is directly dispatched to the event target, and will not bubble nor capture. Applications should not rely on receiving the events listed above during the bubbling or the capturing phase. The third parameter of addEventListener, i.e. "useCapture", will be ignored.

8.2 Extensions to the MediaSynchroniser embedded object

8.2.1 Properties

The following properties are added to the MediaSynchroniser embedded object.

`function onTimeUpdate (Number currentTime, String reason)`	
Description	The function shall be called by the terminal to notify the application of changes in the media when any one or more of the following conditions apply; • The timeline has advanced linearly at the current value of `playbackRate` and no call to the function / event has been posted recently where recently is an implementation specific time between 15 and 250ms. • There has been a discontinuity in the media timeline (as defined in clause 9.2) The `currentTime` argument shall be the value of the `currentTime` property when this event was fired (as defined by `Event.timeStamp`) including the effect of any discontinuity in the media timeline. The `reason` argument shall be "linear" when the function is called due to the first condition listed above (timeline has advanced linearly …). The `reason` argument shall be "other" if the function is called for some other reason. See clause 9.2 for the definition of how this event shall be used with watermark media timelines.

`readonly Number playbackRate`	
Description	Returns the playback rate of the media timeline relative to real-time. If the terminal cannot determine the playback rate then NaN shall be returned. For watermark media timelines, this shall be determined as defined in clause 6.4.2.4.4.

`function onRateChange (Number playbackRate)`	
Description	The function shall be called by the terminal to notify the application of changes in the value of the `playbackRate` property. The function shall be called: (a) when the absolute difference between the current value of the `playbackRate` property and the value of the `playbackRate` property in the most recent call to the `onRateChange` function is greater than 0.1; and (b) when the `playbackRate` property transitions between a numeric value and `NaN` or vice-versa. For example, a transition from normal speed playback to fast forward at a non-supported rate and back again would be reported as two events, one with an argument of NaN and the second with an argument of 1.0. The `playbackRate` argument shall be the value of the `playbackRate` property at the time the event was fired (as defined by `Event.timeStamp`).

8.2.2 Events

When any one or more of the listed conditions apply and no event handler for a timeupdate event is currently executing, the the intrinsic event `onTimeUpdate` shall be generated and a corresponding DOM level 2 event shall be generated in the following manner:

Intrinsic event	Corresponding DOM 2 event	DOM 2 Event properties
onTimeUpdate	timeupdate	Bubbles: No Cancelable: No Context Info: `currentTime, reason`
onRateChange	ratechange	Bubbles: No Cancelable: No Context Info: `playbackRate`

If more than one of the listed conditions applies at the same time and/or one of the listed conditions applies more than once (e.g. more than one discontinuity or more than one change in `playbackRate`), only one timeupdate event shall be generated.

9 System integration

9.1 Use of video/broadcast API and related classes with watermarking

The following shall apply when application discovery using watermarking is employed, once the terminal has obtained a Discovered AIT corresponding to the currently received watermark and an HbbTV® application is running that has or creates a video/broadcast object:

- A `Channel` object shall be created where the values of the properties are taken from of the elements of the same name in the `channel` element in the Discovered AIT except for `ccid`, `dsd` and `terminalChannel`:
  - The value of ccid shall be as specified in the OIPF DAE specification [4].
  - The values of `dsd` and `terminalChannel` are implementation dependent.
  - Reading the `ChannelConfig.channelList` property shall return a `ChannelList` object containing only this `Channel` object.
  - Reading the `ChannelConfig.currentChannel` property, the `currentChannel` property of the video/broadcast object or the `ApplicationPrivateData.currentChannel` property shall return this `Channel` object.
- An `AVAudioComponent` object shall be created for each AVAudioComponent in the Discovered AIT where the values of the properties are taken from the elements of the same name in the corresponding `audioComponent` element in the Discovered AIT.
- An `AVVideoComponent` object shall be created for each AVVideoComponent in the Discovered AIT where the values of the properties are taken from the elements of the same name in the corresponding `videoComponent` element in the Discovered AIT.
- Calls to the `getComponents` method shall return all the AVComponents created using the information from the Discovered AIT, either those of the specified type (if a type was specified in the call to the method) or all of them (if the type specified in the call to the method was null or undefined).
- Calls to the `getCurrentActiveComponents` method shall return one of the AVComponent objects created using information from the Discovered AIT as defined in table 13.

Table 13: Return value for getCurrentActiveComponents method

Watermark State	componentType argument = COMPONENT_TYPE_VIDEO	componentType argument = COMPONENT_TYPE_AUDIO	Comments
Audio watermark detected only	`undefined`	AVComponentCollection containing an AVAudioComponent matching detected audio watermark (see note)	
Audio and unverified video watermarks detected only	`undefined`	AVComponentCollection containing an AVAudioComponent matching detected audio watermark (see note)	
No watermark detected	`undefined`	`undefined`	Applies to applications hidden due to loss of watermark before they are stopped or un-hidden.
Unverified video watermark detected only	`undefined`	`undefined`	
Verified video watermark detected only	AVComponentCollection containing an AVVideoComponent matching detected video watermark (See note 1)	`undefined`	
Audio and verified video watermarks detected	AVComponentCollection containing an AVVideoComponent matching detected audio watermark (See note 1)	AVComponentCollection containing an AVAudioComponent matching detected audio watermark (see note)	
NOTE: The AVComponent matching a detected watermark is the one created from the AVComponent element in the XML AIT whose serverField element is equal to the server code in the detected VP1 watermark. This is the current component as selected in clause 6.4.2.4.2.			

- When watermark state machine changes occur (loss, recovery or change of watermarks within the same service) and the application continues to run, the `onSelectedComponentChanged`/`SelectedComponentChanged` event shall be generated to provide updated component selection information in accordance with table 13 above.

- Calls to the `selectComponent` and `unselectComponent` methods shall fail and no `onSelectedComponentChanged`/`SelectedComponentChanged` event shall be generated.

- While terminals may validate a Discovered AIT against the schema, there is no requirement to do this. XML AITs that fail validation may be considered 'malformed inputs' and discarded by the terminal as defined in clause 9.4.2.

- If the Discovered AIT includes a value that is outside the permitted range for a property then applications may get this value passed to them as is.

- Calling the `bindToCurrentChannel` method for a video/broadcast object in the unrealized state shall succeed and result in the object transitioning through the Connecting state to the Presenting state.

- Calls to the `setChannel` methods shall always fail with an `onChannelChangeError` with error code 11 ("insufficient resources"), except as follows:

  - calls with a value of `null` for the `channel` argument shall transition a broadcast-related application to broadcast-independent state as specified in clause 6.2.2.6.1 of TS 102 796 [1].

  - calls where the `channel` argument is a `Channel` object created from a Discovered AIT shall behave as defined in clause 6.4.4 of the present document.

- Calls to the `prevChannel` and `nextChannel` methods shall always fail with an `onChannelChangeError` with error code 11 ("insufficient resources").
- Calls to the `stop` method shall cause audio and video from the watermarked content source to stop being displayed (while watermark detection from the source content continues).Calls to the `release` method shall have no effect.
- Terminals are not required to support scaling video received over HDMI or may only support it under some circumstances (e.g. support scaling HD but not UHD) (see the scaling attribute in clause 10.2.2). On such terminals, the `video/broadcast` object shall always be in fullscreen mode and hence the following shall apply:
  - The `fullscreen` property shall always return true.
  - The `onFullScreenChange` event shall not be fired.
  - Calls to the `setFullScreen` method with the false argument are silently ignored.
  - The `width` and `height` properties shall be read-only.
- On terminals that do support scaling video received over HDMI, full screen mode and the related APIs listed above shall be supported as specified.
- Reading the `playState` property and the posting of `onPlayStateChange` events shall work as specified.
- If an HbbTV® application is running and the terminal detects a change in the channel received such that the application is allowed to continue running according to the application lifecycle then an `onChannelChangeSucceeded` event shall be fired.
- Calls to the `createChannelObject` methods shall succeed or fail as specified in the description of those methods.

NOTE: Cable, satellite or terrestrial tuners may be disabled when the user selects the HDMI input on a TV set. If this is the case then calls to these methods will fail as if no tuner was present matching the type requested for the channel.

9.2 Use of MediaSynchroniser API with watermarking

Terminals supporting application discovery over broadband using watermarking for an input (e.g. HDMI, broadcast, OTT, IPTV) shall support the use of the watermark media timeline defined in clause 6.4.2.4 as a type of timeline for MediaSynchroniser operations when that input is selected. The timeline selector for the watermark media timeline shall be `"urn:hbbtv:sync:timeline:wm"`. Watermark media time shall be expressed as an integer number of 1 millisecond ticks in the range $[0,2^{53}-1]$.

NOTE: This is JavaScript Number.MAX_SAFE_INTEGER.

If both the terminal and the broadcaster support application discovery over broadband using watermarking, applications shall be able to read the value of `currentTime` using the timeline selector above regardless of how the application was discovered.

Terminals shall support use of the watermark media timeline as the timeline for the master media of the `MediaSynchroniser` object when the master media is a video/broadcast object associated with presentation of content from which the terminal is performing application discovery using watermarking. When this applies, the extensions to the `MediaSynchroniser` object defined in clause 8.2 shall be supported with the following additional requirements;

- The `playbackRate` shall be calculated as defined in clause 6.4.2.4.4.
- The `currentTime` argument shall be calculated according to clause 6.4.2.4.4, and shall take into account any discontinuity in watermark Interval Code or time_offset values in received VP1 messages.
- A discontinuity in the watermark media timeline shall be reported when the difference between the expected value and the actual value is more than $1/30^{th}$ of a second and shall not be reported for changes less than this.

NOTE: Some jitter in the watermark media timeline will happen when it is derived from the video watermark and time_offset values are used in VP1 messages. The above requirements expose this to applications but do not generate `timeupdate` events only because of this jitter.

This clause extends the functionality specified in clause 13.4 of TS 102 796 [1] (Timelines and timestamping), enabling synchronization of applications to watermarked media (e.g. via access to the currentTime property of a MediaSynchroniser object).

The present document does not require support for use of the watermark media timeline in the context of either multi-stream synchronization or inter-device synchronization.

9.3 Use of Stream Events API

9.3.1 Stream events in the presence of DVB Service Information

When the broadcast does not carry an AIT, then it is also unlikely that other broadcast-carried HbbTV® functions like application transport with DSM-CC, stream events, etc., will be available in the video broadcast. Methods and properties related to these functions will be handled the same way as in TS 102 796 [1].

Note also that HbbTV® applications that are discovered using the method described in the present document can be tailored to the discovery method, so that they can take into account - and work around - the inherent limitations of this method.

9.3.2 Stream events with application discovery using ATSC3 watermarking

9.3.2.1 General

Applications where `Application.lifecycleControl` is "xmlait-atsc3" shall be able to register for stream events and receive stream events when detected according to clauses 9.3.2.2 and 9.3.2.3. This applies regardless of whether an application is an autostart application whose launch was initiated by the HbbTV® terminal or a present application whose launch was initiated by another HbbTV® application.

Broadcast-independent applications (including ones that were previously broadcast-related but changed) shall not be able to register for stream events and registrations shall be cancelled on changing from broadcast-related to broadcast-independent.

9.3.2.2 Events delivered in video watermarks

The dynamic_event_message() defined in ATSC A/336 [9] supports delivery of stream events in video watermarks that conform to the structure of events used in ATSC 3.0 broadcast delivery protocols. For HbbTV®, the A/336 protocol is extended to define the syntax and bitstream semantics of the Video Watermark Event Message as given in table 14 and the parameter descriptions that follow.

Table 14: Bit Stream Syntax for the HbbTV® Dynamic Event Message

Syntax	No. of Bits	Format
dynamic_event_message() {		
delivery_protocol_type	4	uimsbf
reserved	4	'1111'
if (delivery_protocol_type == '3') {		
event_name_length (N1)	8	uimsbf
event_name	8*N1	
data_length (N2)	8	uimsbf
data	8*N2	
}		

delivery_protocol_type: This 4-bit field shall signify the service to which the Dynamic Event applies. Table 15 describes the encoding of this field.

Table 15: delivery_protocol_type field Encoding

delivery_protocol_type	Meaning
3	HbbTV®

event_name_length: This 8-bit integer shall give the length of the event_name field in bytes.

event_name: This field shall convey the name of the stream event.

data_length: This 8-bit integer shall give the length of the data field in bytes.

data: This field shall contain the data of the stream event.

The following constraints apply:

- The values of the **event_name_length** and **data_length** fields shall each be less than or equal to 256 (bytes).

When the watermark state is either "Verified video watermark detected only" or "Audio and verified video watermarks detected", the terminal shall deliver Video Watermark Event Messages present in video watermarks to broadcaster applications using the DSM-CC stream event API specified in clause 8.2.1 of TS 102 796 [1] upon their detection. Video watermark event messages shall be discarded without being delivered if the watermark state is either "Audio and unverified video watermarks detected" or "Unverified video watermark detected" or if the contents of the event_name in the dynamic event message do not match the eventName argument in a call to addStreamEventListener.

The targetURL associated with the video watermark event stream shall be `urn:hbbtv:streamevent:a336:video`. The **event_name** and **data** fields of the Video Watermark Event Message shall be used by the terminal as the source for populating the `name` and data elements of the DSM-CC `StreamEvent` class. The `text` property of the StreamEvent object shall contain the contents of the data field of the Video Watermark Event Message as a string assuming UTF-8 as the encoding. Bytes that cannot be transcoded into a Unicode character are skipped.

Application developers should be aware that in some circumstances an attacker may be able to modify the video signal including the data in the Video Watermark Event Message. Applications shall not use this data in a way that would result in it being executed by the browser. Applications should be written to be tolerant of incorrectly formatted data or values which are outside the expected range without hanging up or crashing.

For purposes of determining uniqueness of detected triggers, the terminal shall consider successively detected Video Watermark Event Messages as repeated instances (and deliver only one instance to the application) if the contents of their wm_message_block (see clause 5.1 of ATSC A/336 [9]**)** are identical.

StreamEvents with status "error" shall be dispatched when any of the following conditions apply:

- The terminal does not support monitoring for video watermarks in the content currently being presented by the video/broadcast object.

EXAMPLE 1: Terminals that do not support monitoring for video watermarks at all.

EXAMPLE 2: Terminals that support monitoring for video watermarks in content received over HDMI that do not support monitoring for video watermarks in content received over classical broadcast, over IPTV or over broadband when the video/broadcast object is presenting the latter content.

- The watermark state (see clause 6.2.1) changes from "Audio and verified video watermarks detected" to "Audio watermark detected only" or from "Verified video watermark detected only" to "No watermark detected".

NOTE: If the "loss of watermark" process is followed and the application continues to run then it will need to re-register for any stream events that are still appropriate.

The following conditions that are conceptually similar to a required error condition when the API is used with MPEG-2 transport streams shall not generate an error when used with watermarks as defined in this clause:

- The watermark state is currently "Audio watermark detected only".

- The `eventName` argument passed to the `addStreamEventListener` method cannot be found. (Terminals do not have a list of video watermark event names in use by a broadcast service).

9.3.2.3 Deriving stream events from VP1 payloads

Applications shall be able to register to be notified when the value of the `query_flag` field changes in VP1 Payloads using the DSM-CC stream event API specified in clause 8.2.1 of TS 102 796 [1].

The target URL associated with the watermark `query_flag` event stream shall be `urn:hbbtv:streamevent:a336:audio`. The `eventName` argument shall be ignored.

When the terminal determines that a state change in the Query Flag has occurred in accordance with the method specified in clause 6.4.2.1, a StreamEvent object shall be delivered to an application that has registered using the `addStreamEventListener` method as follows.

- The `name` property of the StreamEvent object shall contain the value of the server_field from the VP1 payload encoded as a decimal number.

NOTE 1: The above requirement is an intentional significant deviation from when this API is used with DSM-CC stream events from an MPEG-2 transport stream.

- The `data` property of the StreamEvent object shall contain the entire VP1 playload as defined in clause 5.2.3 of ATSC A/336 [9] encoded in hexadecimal.
- The `text` property of the StreamEvent object shall contain "".
- The `status` property of the StreamEvent object shall be as defined in TS 102 796 [1]. `StreamEvent`s with status "error" shall be dispatched when any of the following conditions apply.
  - The terminal does not support monitoring for audio watermarks in the content currently being presented by the video/broadcast object.

EXAMPLE 1: A terminal supporting the present document that does not support monitoring for watermarks in content delivered by classical broadcast, by IPTV or by broadband when a video/broadcast object is presenting such content.

  - The watermark state (see clause 6.3.1) changes from "Audio and verified video watermarks detected" to "Verified video watermark detected only" or from "Audio watermark detected only" to "No watermark detected".

EXAMPLE 2: If a query_flag change has been identified in a VP1 payload with domain_type equal to 0, server_field value equal to 1074976391, interval_field value equal to 7615, and query_flag equal to 1, then a `StreamEvent` object would be provided with `name` property equal to "1074976391", a `data` property equal to "1004B5A1C3B7F", a `text` property equal to "", and `status` property equal to "trigger".

The following conditions that are conceptually similar to a required error condition when the API is used with MPEG-2 transport streams shall not generate an error when used with watermarks as defined in this clause:

- The watermark state is "Verified video watermark detected only" or "Unverified video watermark detected" or "No watermark detected".

EXAMPLE 3: The content presented by the video/broadcast object is received over broadcast (not HDMI) and the terminal supports detecting watermarks in content received over broadcast.

EXAMPLE 4: The audio watermark has been temporarily lost but the application continues to run under the control of a verified video watermark.

NOTE 2: The definition of `query_flag` in ATSC A/336 [9] says that "*A change in the value of this field between successive VP1 Payloads in a VP1 Audio Watermark Segment or between successive VP1 Message Groups in a VP1 Video Watermark Segment indicates that a Dynamic Event (as defined in A/337 [*6*]) is available from the Dynamic Event HTTP server.*" In the present document, it is the HbbTV® application that is responsible for fetching something from a server and not the HbbTV® terminal. What is fetched from the server may be anything mutually agreed between the HbbTV® application and the server (which may be operated by the same organization).

For purposes of determining uniqueness of detected triggers, the terminal shall consider each VP1 Payload detected from the audio watermark that conveys a change in the value of `query_flag` from the value detected in the previous

VP1 Payload in the Audio Watermark Segment to indicate a unique event. No event shall be reported if `query_flag` does not change.

9.4 Reliability and resilience

9.4.1 User interaction

Terminals shall operate reliably in response to (rapid) user interaction. Terminals shall keep operating the application discovery over broadband mechanisms reliably in the following events:

- For streams obtained from HDMI:
  - Rapid appearance and disappearance of the video watermark (e.g. equivalent to opening and closing an STB interface or signal loss) where the audio watermark remains.
  - Rapid appearance and disappearance of the video and audio watermark (e.g. equivalent to opening and closing an STB interface or signal loss).
    - Where the same server code is found when the watermark returns (e.g. equivalent to muting and unmuting the audio on the STB).
    - Where a different server code is found when the watermark returns (e.g. equivalent to changing the channel on the STB).
  - Rapid appearance and disappearance of the audio watermark while the video watermark remains.
  - Loss and restoration of HDMI input (e.g. user turns the set-top box off and on again).
- For streams obtained using a (regular) tuner:
  - Repeatedly switching between channels with a DNS entry corresponding to the DVB Service Information, and no watermarks and channels with no DNS entry corresponding to the DVB Service Information and no watermarks.
  - Repeatedly switching between channels with a regular AIT and DVB Service Information to channels with DVB Service Information only.

9.4.2 Malformed or malicious inputs

Terminals shall be resilient to malformed, malicious inputs and/or operational errors that are likely to occur. Specifically, in the following circumstances, the terminal shall discard any malformed inputs and remain responsive to valid inputs in the future, including starting and stopping applications:

- Failed DNS CNAME requests for requests made according to clause 5.2.
- DNS CNAME responses referring to non-existing servers for requests made according to clause 5.2.
- Incomplete AITs (e.g. due to server errors) after a successful HTTP download.
- Unavailable AITs (e.g. HTTP errors 404 or 500), for example caused by:
  - TLS authentication/verification failure.
  - Operational errors.
- When fetching an AIT larger than the supported size (see clause 10.2.1), the terminal is not expected to correctly download and/or process the AIT.

Specifically, terminals shall remain responsive to channel change, application termination requests, and remain able to discover applications over broadband according to clause 5 in the following circumstances:

- Loss of watermark during DNS requests initiated according to clause 5.2.

- Loss of watermark while fetching an AIT as specified in clause 6.4.2.2.

9.4.3 Long-term use

On the application level, terminals shall reliably handle the following uses:

- An application rapidly switching back and forth between broadcast-independent and broadcast-related modes 200 times.

Terminals shall be resilient to transient error conditions that are likely to occur, as well as to conditions of low resource availability. Specifically, the terminal shall remain responsive to channel change, application termination requests in the following circumstances:

- On overflows in the watermark interval field, the continuity of the lifecycle of any running application is not expected.
- Discovering a new service, when the channel list is already full.

On power loss, terminals shall operate according to the present document where specified when power is returned. Specifically:

- AIT downloads which are in progress on loss of power shall not have an impact on the terminals ability to perform discovery of applications over broadband.

9.4.4 Watermarks and XML-AIT

Terminals shall remain able to discover applications over broadband according to the present document when:

- the VP1 payload query_flag field value is switched 200 times in a row;
- there are 200 discontinuities in the VP1 payload Interval Code field;
- the VP1 payload Server Code is changed 200 times in a row.

and also when:

- either or both of the mediaTimeAnchor or the intervalFieldAnchor field in the XML-AIT are changed 200 times in a row.

9.5 Presenting HDMI-delivered video, broadband-delivered video and switching between them

The `broadbandOverlay` attribute in the XML capabilities (see clause 10.2.2) indicates if broadband delivered video (e.g. an advert) can overlay video delivered via HDMI (true) or not (false). The following apply to terminals where the value of this attribute is false.

- HDMI-delivered video will start being presented when the user selects an HDMI input on the UI of a TV set. This is outside the scope of the present document.
- HDMI-delivered video will stop being presented when the user changes the selected input on the UI of a TV set away from an HDMI input to something else. This is outside the scope of the present document.
- HDMI-delivered video shall stop being presented when any of the following happen;
    - When an application calls the `stop` method on a video/broadcast object that is in the presenting state and is associated with presentation of content from HDMI.
    - When a broadcast-related application whose lifecycle is controlled by watermarks covers the entire screen with opaque graphics
    - When a broadcast-related application whose lifecycle is controlled by watermarks changes to become broadcast-independent

- When a call to `switchMediaPresentation` from HDMI-delivered content to broadband-delivered content is successful.

- HDMI-delivered video shall start being presented when any of the following happen;
  - When an application calls the `bindToCurrentChannel` method on a video/broadcast object that is in the stopped state and is associated with presentation of content from HDMI.
  - When a broadcast-related application whose lifecycle is controlled by watermarks exits
  - When a broadcast-related application whose lifecycle is controlled by watermarks that 1) was covering the entire screen with opaque graphics and 2) is not presented video delivered by broadband stops doing both.
  - When a call to `switchMediaPresentation` from broadband-delivered content to HDMI-delivered content is successful.
- Presentation of video shall switch from HDMI-delivered video to broadband-delivered video when an broadcast-related application does the following;
  - Obtains a video/broadcast object associated with presentation of content from HDMI
  - Calls the `stop` method on that video/broadcast object
  - Waits for the state of the video/broadcast object to change to stopped
  - Calls `play` on an HTML5 video element whose `source` or `src` is an http: URL, an https: URL or a URL returned by calling `createObjectURL` with a `MediaSource` as the argument

  Any visible graphics from the HbbTV application shall be visible at all times.
- Presentation of video shall switch from broadband-delivered video to HDMI-delivered video when a broadcast-related application does the following;
  - Follows the current best practice defined in Annex J of TS 102 796 [1] for ensuring that all references to a media element are removed when removing it from the DOM.
  - Calls the `bindToCurrentChannel` method on a video/broadcast object in the stopped state

  Any visible graphics from the HbbTV application shall be visible at all times.

The `broadbandOverlay` attribute in the XML capabilities (see clause 10.2.2) indicates if broadband delivered video (e.g. an advert) can overlay video delivered via HDMI (true) or not (false). If a broadcast-related application calls the `play`() method on an HTML5 video element that is in front of a video/broadcast object in the presenting state which is in turn associated with the presentation of content from HDMI then the following shall apply;

- Regardless of the value of the `broadbandOverlay` attribute, the HDMI video shall continue to be presented without interruption and the video/broadcast object shall continue in the presenting state.
- If the value of the `broadbandOverlay` attribute is false then the play request shall fail and the "dedicated media source failure steps" defined for the HTML5 video element shall be followed.
- If the value of the `broadbandOverlay` attribute is true then the play request on the HTML5 video element shall proceed and shall not fail due to a resource conflict with the video/broadcast object. If the request does not fail for some other reason then both the video from the HTML5 video element and from the video/broadcast object shall be visible.

9.6 Targeted advertising

9.6.1 Terminal capabilities for targeted advertising

Terminals supporting the combination of targeted advertising (as defined in TS 103 736-1 [15]) and application discovery over broadband using watermarks (as defined in the present document) in services received via HDMI shall support the following;

- Using a video/broadcast object representing video received via HDMI in the calls to the `switchMediaPresentation` method in the fast media switch API;
  - As the `originalMediaObject` argument when `newMediaObject` is an HTML5 video element
  - As the `newMediaObject` argument when `originalMediaObject` is an HTML5 video element
  - Use of `urn:hbbtv:sync:timeline:wm` as the `timelineSelector` argument to refer to the watermark media timeline (as defined in clause 6.4.2.4 of the present document) when either of the previous apply
  - Switching from content received via HDMI to content delivered by broadband and back again as directed by an HbbTV application calling the `switchMediaPresentation` method
- As defined in clause 10.1 of the present document, either;
  - Displaying broadband delivered video simultaneously with video received via HDMI or
  - Switching between displaying video received via HDMI and video received via broadband and vice-versa
- Performing the following operations 1) while content received via HDMI is being displayed, 2) when the lifecycle of a running HbbTV application is controlled by either a video watermark or an audio watermark (even if no content that is received via HDMI is being displayed and even if broadband delivered video and audio are being presented) and 3) when both the previous conditions apply;
  - Detecting watermarks in the content received via HDMI and controlling the lifecycle of an HbbTV application
  - Establishing and maintaining the watermark media timeline

9.6.2 Behaviour of the Fast Media Switch API with watermarks

The watermark media timeline has units of milliseconds and a limit of 2^{53} ticks. The requirements for comparisons and evaluation of inequality relationships specified in clause 4.2 of TS 103 736-1 [15] with respect to values on MPEG PTS and TEMI timelines shall also apply to the watermark media timeline.

The following preconditions shall be added to step 2 of clause 8.1.4.1 of TS 103 736-1 [15]:

- If `originalMediaObject` is a video/broadcast object representing video received via HDMI, then the following is met:
  - the terminal supports the requirements listed in clause 9.6.1 of the present document; and
  - the capability `monitoringAWMWhilePlayingBroadband` (see clause 10.2.2) of the terminal is `true`

The following steps shall be performed between steps 1 and 2 of the "executing the switch" algorithm specified in clause 8.1.4.3 of TS 103 736-1 [15]:

1.1) If the `originalMediaObject` is a video/broadcast object representing video received via HDMI and one or more watermark states were specified in the `permittedWatermarkStates` argument and the current watermark state (`oipfApplicationManager.watermarkState`) is not one of the states so specified then reject `promise` with a `InvalidStateError`.

1.2) If the `originalMediaObject` is a video/broadcast object representing video received via HDMI and the playback rate (i.e. what would be the value of the `playbackRate` property of a Media Synchroniser) is not between `0.99` and `1.01`, then reject `promise` with a `InvalidStateError`.

10 Capabilities

10.1 Display model

Video received via HDMI shall be displayed in the "Video plane" in the logical plane model defined in clause H.1 of the OIPF DAE specification [4]. Terminals shall support displaying HbbTV application graphics in the “DAE application graphic plane”of that model, overlayed on top of video received via HDMI. Terminals may support displaying video received by broadband and video received by HDMI simultaneously in which case the video received by broadband shall appear in front of the video received by HDMI and behind any subtitles.

Terminals supporting the combination of targeted advertising (as defined in TS 103 736-1 [15]) and application discovery over broadband using watermarks in services received via HDMI shall support either;

- displaying video received by broadband and video received by HDMI simultaneously as previously mentioned; or
- switching between displaying video received via HDMI and video received via broadband and vice-versa

10.2 Terminal capabilities and functions

10.2.1 Minimum terminal capabilities

Terminals shall support XML AIT files whose size is less than or equal 256 KB. Terminals may support XML AIT files larger than this.

10.2.2 HbbTV® reported capabilities and option strings

The support of application discovery over broadband shall be indicated by the addition of one or more `<applicationDiscovery>` elements in the XML device capabilities as defined in clause A.2.1. For example:

```
<applicationDiscovery>urn:hbbtv:discovery:atsc3audio</applicationDiscovery>
<applicationDiscovery>urn:hbbtv:discovery:atsc3video</applicationDiscovery>
<applicationDiscovery>urn:hbbtv:discovery:dvbsi</applicationDiscovery>
```

Support for the application discovery based on DVB-SI as defined in clauses 5.3.1, 5.4.1 and 5.6.1 of the present document shall be indicated by `urn:hbbtv:discovery:dvbsi`.

Support for the application discovery based on the ATSC3 video watermark as used in the present document shall be indicated by `urn:hbbtv:discovery:atsc3video`. Support for the ATSC3 audio watermark as used in the present document shall be indicated by `urn:hbbtv:discovery:atsc3audio`.

NOTE 1: Clause 6.3.3 includes a requirement that `urn:hbbtv:discovery:atsc3audio` be indicated if `urn:hbbtv:discovery:atsc3video` is indicated but not vice-versa.

Support for running applications in combination with video and audio received via HDMI shall be indicated by the addition of a single `<hdmi>` element in the XML device capabilities as defined in clause A.2.1. For example:

```
<hdmi broadbandOverlay="false" monitoringAWMWhileBroadband="true" scaling="true"/>
```

The `broadbandOverlay` attribute shall indicate if broadband delivered video can overlay video delivered via HDMI (`true`) (e.g. using broadband delivered video as picture in picture overlaying HDMI video) or not (`false`) (i.e. display of HDMI video stops when broadband delivered video is shown). See clause 9.5 for related requirements.

The `monitoringAWMWhilePlayingBroadband` and `monitoringVWMWhilePlayingBroadband` attributes shall indicate if monitoring HDMI delivered content for audio and video watermarks respectively is supported while broadband delivered video and audio is being presented (`true`) or not (`false`).

NOTE 2: If both these attributes are false then applications may need to make themselves broadcast-independent prior to playing broadband-delivered video in order to avoid being stopped.

The `scaling` attribute shall indicate if video received by HDMI can be scaled to fit into a video/broadcast object that is not in fullscreen mode (`true`) or not (`false`). If scaling is supported but with restrictions that are not compatible with those for "video scaling" in clause 10.2.1 of TS 102 796 [1] then `false` shall be returned.

Terminals shall include an element of the following form to describe current audio output capabilities of the terminal:

```
<audio_system audio_output_format="audio-output-channels"
    audio_output_interface="audio-output-type" hdmi_audio_pcm=”xs:boolean”/
    audio_system_between_TV_and_STB=”xs:boolean”>
```

NOTE 3: This element is also included in version 1.6.1 of TS 102 796 [1] but only with the audio_output_format attribute. The other attributes are added in the present document.

Where the attributes shall be as follows;

- `audio_output_format` – defines the terminal handling of multi-channel audio as defined below.
- `audio_output_interface` – defines how the terminal audio is output to the user as defined below
- `hdmi_audio_pcm` – if audio_ouput_interface is “`hdmi-arc-sac`” then this shall indicate if the audio format on the HDMI interface is pcm (`true`) or bitstream (`false`).
- `audio_system_between_TV_and_STB` – if `audio-output-interface` is “`hdmi-arc-sac`” then this shall indicate if the audio output device is between the TV and the STB or not.

NOTE: If the audio output device is between the TV and the STB then switching the audio output device to use the audio from the TV may result in the audio from the STB no longer reaching the TV. If this happens then the loss of watermark behaviour may be triggered on the TV resulting in the interruption of the presentation of the advert.

Where `audio-output-channels` is a string containing one of the following:

- `stereo` – the terminal and any connected devices known to it only have active stereo audio outputs; any multi-channel audio sources will be downmixed. An expertly mixed stereo audio source can be expected to give a better audio experience than a multi-channel source, which the terminal would just downmix to stereo.
- `multichannel` – the terminal has one or more active multi-channel audio outputs and multi-channel audio is enabled by terminal settings; multi-channel audio sources should be heard without downmixing to stereo. The terminal may be outputting multi-channel audio directly or sending it to an external device which supports the format.
- `multichannel-preferred` – the conditions for “multichannel” do not apply but the terminal believes that it can produce a better audio experience if the application provides a multi-channel audio source than if the application provides an expertly mixed stereo source.

NOTE: The definition of `audio-output-channels` is identical to that in version 1.6.1 of TS 102 796 [1].

Where `audio-output-type` is a string that shall contain one of the following;

- `local` – if the terminal audio output is via speakers integrated into the TV or speakers directly connected to the TV via a point to point connection (including Bluetooth) or a headphone socket on the TV.
- `hdmi-arc-sac` - if the terminal audio output is using HDMI Audio Return Channel or HDMI Enhanced Audio Return Channel, and HDMI System Audio Control is supported to switch the audio output device to use the audio from the TV and back to use audio from the STB.
- `disabled` - if terminal local audio output is permanently disabled. Examples of this shall include 1) the user disabling local audio using a setting in the terminal UI (where such a setting exists) or 2) the user permanently setting terminal audio volume to zero or 3) the user permanently muting terminal audio.

- `other` – if none of the above apply, e.g. for terminal audio outputs like S/PDIF (where the terminal cannot know whether what is connected is speakers or an audio system where the STB may also be connected), e.g. HDMI when CEC is not supported.

The electronic attachments to the present document (see annex B) include the XML schema for the capabilities and an example which using the extended capabilities that validates using that schema.

10.2.3 Performance profiles

The present document defines 3 performance profiles for use in the `minimumSwitchPerformanceRequired` argument of the `switchMediaPresentation` method when the present document and TS 103 736-1 [15] are used in combination.

URN	broadcast-to-broadband accuracy limit (see clause 4.2.3 of TS 103 736-1 [15])	broadcast-to-broadband duration (see clause 4.2.4 of TS 103 736-1 [15])		broadband-to-broadcast accuracy limit (see clause 4.2.5 of TS 103 736-1 [15])	broadband-to-broadcast duration (see clause 4.2.6 of TS 103 736-1 [15])	
	A1	D1min	D1max	A2	D2min	D2max
urn:hbbtv:ta-hdmi:profile:2020:1 (1)(3)	40	0	240	40	0	240
urn:hbbtv:ta-hdmi:profile:2020:2 (2)(3)	40	0	120	40	0	120
urn:hbbtv:ta-hdmi:profile:2020:3 (4)	40	0	40	40	0	40

NOTE 1: These values are identical to those for profile 1 in TS 103 736-2 [15]. That document explains that the requirements for profile 1 are "based on a landing period of 0,5 s to 0,6 s in addition to a GOP length of 1,5 s. Such a landing period is believed to be more acceptable at the end of an advert break than in the middle of one."

NOTE 2: These values are indentical to those for profile 2 in TS 103 736-2 [15]. That document explains that the requirements for profile 2 are "based on a landing period of 4 frames (at 25 Hz) between each ad. This may be because a broadcaster inserts some black frames between individual ads. Alternatively it may be technically and commercially possible to omit the first and last frames of some adverts."

NOTE 3: Intended for terminals that cannot display video from broadband and video from HDMI simultaneously and that need to reconfigure part of the video pipeline to switch between them.

NOTE 4: This profile is intended for terminals that can display video from broadband and video from HDMI simultaneously. Displaying a full screen advert would fully obscure the video from HDMI.

If the terminal meets the requirements for urn:hbbtv:ta-hdmi:profile:2020:1 then the XML capabilities shall include a `profile` element (see clause 10.3.1 of TS 103 736-1 [15]) containing that URN. Otherwise such a `profile` element shall not be included.

If the terminal meets the requirements for urn:hbbtv:ta-hdmi:profile:2020:2 then the XML capabilities shall include at least two `profile` elements, one for that URN and one for urn:hbbtv:ta-hdmi:profile:2020:1. If the terminal does not meet the requirements for urn:hbbtv:ta-hdmi:profile:2020:2 then a `profile` element containing that URN shall not be included.

If the terminal meets the requirements for urn:hbbtv:ta-hdmi:profile:2020:3 then the XML capabilities shall include `profile` elements for at least three URNs, one for that URN, one for urn:hbbtv:ta-hdmi:profile:2020:1 and one for urn:hbbtv:ta-hdmi:profile:2020:2. If the terminal does not meet the requirements for urn:hbbtv:ta-hdmi:profile:2020:3 then a `profile` element containing that URN shall not be included.

Profile elements for profiles defined in the present document shall be included after `profile` elements for profiles defined in TS 103 736-2 [16].

11 Security

11.1 Introduction

Application discovery over broadband has been designed to limit risks posed by bad actors. Some known security risks are listed here and recommendations for their mitigation in deployments provided.

11.2 Modification of DVB Service Information or the VP1 Payload of the watermark

11.2.1 Risks

When an application is retrieved via broadband, the terminal determines the hostname of the Broadcaster DNS Root via DNS based on either DVB Service Information received from the tuner or the server_field of a VP1 payload retrieved from an audio watermark.

Neither received DVB service info nor detected VP1 payload data can be authenticated by the terminal. As a result, if an intermediary has modified the information to convey values that identify a different service for which application discovery is supported, an application associated with a different service from the one being received will be discovered. If it is modified to values for which application discovery is not provided (or is removed altogether), then application discovery will fail.

The potential for harm as a result of such modifications is limited by the fact the universe of Broadcaster DNS Root server hostnames that can be identified is restricted to those listed in the hbbtvdns.org authoritative nameserver records. Modification of DVB Service Information or VP1 payload data cannot result in retrieval of a Broadcaster DNS Root server hostname outside of this universe.

11.2.2 Applicable Application Discovery Methods

- Application Discovery using DVB Service Info
- Application Discovery using Watermarking

11.2.3 Mitigation Techniques

The risk of DVB service info or VP1 payload data modification causing a malicious application being discovered can be mitigated by ensuring that the operational processes of the administrator of the hbbtvdns.org domain provide adequate safeguards to prevent the inclusion of DNS records that identify compromised or unauthorized Broadcaster DNS Root servers.

11.3 Modification of dynamic event messages carried in watermarks

11.3.1 Risks

During application discovery using watermarks, dynamic event messages carried in the watermarks may be delivered to running broadcaster applications. The watermark-based dynamic event message delivery protocol does not provide message authentication by the terminal. If an intermediary has modified the dynamic event message, the application may receive a different dynamic event message from that which was transmitted or no dynamic event message at all.

11.3.2 Applicable Application Discovery Methods

- Application Discovery using Watermarking

11.3.3 Mitigation Techniques

Broadcasters can mitigate the risk of dynamic event message modification by an intermediary by implementing dynamic event message authentication in in their HbbTV® application. Methods for dynamic event message authentication include carriage of message authentication codes in dynamic events message watermarks or alternately implementation of a message authentication protocol between the broadcaster application and a broadcaster server.

11.4 Attacks on DNS Resolution

11.4.1 Risks

When an application is retrieved via broadband, the terminal discovers the hostname of the Broadcaster DNS Root server and AIT server using DNS resolution. The specified DNS resolution protocol does not by provide a mechanism for the terminal to authenticate the hostnames received via this protocol in a manner that would protect against receiving malicious hostnames from a compromised DNS recursive resolver or a man-in-the-middle of the DNS resolution process. This can lead to terminals retrieving XML AITs for malicious applications.

11.4.2 Applicable Application Discovery Methods

- Application Discovery using DVB Service Info
- Application Discovery using Watermarking

11.4.3 Mitigation Techniques

The HbbTV® organization can mitigate the risk of compromised DNS resolution by ensuring that the administrator of the hbbtvdns.org domain implement the DNSSEC protocol (IETF RFC 4033 [i.4], IETF RFC 4034 [i.5], IETF RFC 4035 [i.6]).

Terminal manufacturers are advised to implement DNSSEC authentication (IETF RFC 4033 [i.4], IETF RFC 4034 [i.5], IETF RFC 4035 [i.6]) of DNS resource records received during broadcaster DNS Root server and AIT server hostname resolution.

Terminals may employ XML DSIG (as specified in clause 7.1.1) to allow rejection of any unauthorized XML AITs retrieved from an attacker-controlled AIT server. The certificates and/or keys to validate such signatures are outside the scope of the present document.

12 Privacy

12.1 Introduction

Application discovery over broadband has been designed to protect the privacy of the end user. Service information and server addresses are generally stored within the terminal and DNS transactions are performed asynchronous to viewing activity in order to reduce information observable by third-parties. The information that is potentially disclosed to channel providers (broadcasters) on using a Discovered AIT is comparable to what is disclosed when using an AIT signalled in the broadcast.

Some known privacy risks are listed here and recommendations for their mitigation in deployments provided.

12.2 Application Retrieval via Broadband

12.2.1 Risks

When an application is retrieved via broadband, the terminal resolves the hostname of the application server using DNS and retrieves the application from the application server using HTTP over TLS.

If the hostname of an application server is uniquely associated with a single broadcast service, the DNS resolution of the hostname by the terminal could be used by the DNS resolver operator or a network eavesdropper to determine that the terminal is viewing the service.

Similarly, if the IP address of an application server is uniquely associated with a single broadcast service, the HTTP over TLS traffic between the server and the terminal could be used by the network eavesdropper to determine that the terminal is viewing the service.

12.2.2 Applicable Application Discovery Methods

- Application Discovery using Broadcast AIT
- Application Discovery using DVB Service Info
- Application Discovery using Watermarking

12.2.3 Mitigation Techniques

Broadcasters can mitigate this risk by avoiding the use of hostnames or IP addresses with application servers that are uniquely associated with a single broadcast service.

12.3 AIT Retrieval via Broadband

12.3.1 Risks

When an AIT is retrieved via broadband, the terminal resolves the hostname of the AIT server using DNS and retrieves the AIT from the AIT server using HTTP over TLS.

If the hostname of an AIT server is uniquely associated with a single broadcast service, the DNS resolution of the hostname by the terminal could be used by the DNS resolver operator or a network eavesdropper to determine that the terminal is viewing the service.

Similarly, if the IP address of an AIT server is uniquely associated with a single broadcast service, the HTTP over TLS traffic between the server and the terminal could be used by the network eavesdropper to determine that the terminal is viewing the service.

12.3.2 Applicable Application Discovery Methods

- Application Discovery using DVB Service Info
- Application Discovery using Watermarking

12.3.3 Mitigation Techniques

Broadcasters can mitigate this risk by avoiding the use of hostnames or IP addresses for AIT servers that are uniquely associated with a single broadcast service.

12.4 Authoritative FQDN Resolution Using Watermarking

12.4.1 Risks

Authoritative FQDN resolution using data from watermarks is generally performed asynchronously to the viewing of the service carrying that watermark. However, the first time a terminal detects a watermark conveying a specific server_field value, it will perform DNS record resolution using that value. The DNS resolution of the Authoritative FQDN by the terminal could be used by the DNS resolver operator or a network eavesdropper to determine that the terminal is viewing a particular service.

12.4.2 Applicable Application Discovery Methods

- Application Discovery using Watermarking

12.4.3 Mitigation Techniques

Broadcasters can mitigate this risk by using the large domain format of the VP1 payload and changing server_field values in their broadcast service infrequently (e.g. not more than once per calendar year).

Annex A (normative): OIPF specification profile

A.1 Detailed section-by-section definition for volume 5

Table A.1 defines clauses of the OIPF DAE specification [4] that are required by the present document but which are either not required at all by TS 102 796 [1] or where the present document introduces additional requirements beyond what is required by the OIPF DAE specification [4]. Where a class or object is partly required by TS 102 796 [1], the properties and/or methods and/or events required by [1] are required by the present document. Only additional requirements are listed here. Methods properties and events that are not required by TS 102 796 [1] and not required by the present document should not be supported unless required by another specification.

Table A.1: Section-by-section profile of the OIPF DAE specification

Section, sub-section	Reference in DAE [4]	Status in HbbTV®	Status in the present document
The application/oipfApplicationManager embedded object	7.2.1	M(*)	The `watermarkState` property defined in clause 8.1 shall be supported.
The Application class	7.2.2	M(*)	The `lifecycleControl` property defined in clause 8.1 shall be supported.

The LocalSystem class	7.3.3	NI	Read-only access to the following properties shall be supported:
			`mute`
			`volume`
			Clause A.2.2 shall be supported.
			Note that operator applications as defined in TS 103 606 [i.8] are required to have read-write access to this property under circumstances defined in that document. That requirement shall only apply to operator applications and not for regular HbbTV® applications.

A.2 Modifications, extensions and clarifications to volume 5

A.2.1 Extensions to the OIPF-defined capability negotiation mechanism

The following schema is used instead of the schema defined by annex F of the OIPF DAE specification [4]. The normative definition of this schema is found in the electronic attachments - see annex B of the present document.

NOTE: This schema requires XML schema version 1.1.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:termcap="urn:hbbtv:config:oitf:oitfCapabilities:2017-1"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:vc="http://www.w3.org/2007/XMLSchema-versioning"
           targetNamespace="urn:hbbtv:config:oitf:oitfCapabilities:2017-1"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified"
           vc:minVersion="1.1"
           version="2021.1">

    <xs:annotation>
        <xs:documentation>This is the schema for the HbbTV XML Capabilities. This schema is authored
to validate using tools supporting v1.1 of the XML schema specification. Tools only supporting v1.0
will not be able to validate using this schema without modifying it.</xs:documentation>
    </xs:annotation>

    <!-- ************************************************************** -->
    <!-- Start CEA-with-OIPF-and-HbbTV-mods -->


    <xs:redefine schemaLocation="ce-html-profiles-1-0.xsd">


        <xs:complexType name="profileListType">
            <xs:complexContent>
                <xs:extension base="termcap:profileListType">
                    <xs:sequence>
                        <!-- Introduced in HbbTV 2.0.1 -->
                        <!-- NOTE: This element was defined in the original HbbTV 2.0.1
                             schema as going in the <ext> tag, not here.  However, the
                             original HbbTV 2.0.1 specification text included an
                             example that put it here.  Hence this fixed schema allows
                             it in either place.  For best compatibility, applications
                             should look for it in both places.
                             -->
                        <xs:element name="html5_media" type="xs:boolean" minOccurs="0"/>

                        <!-- NOTE: This element was defined in the original OIPF/HbbTV
                             schema as going in the <ext> tag, not here.  However, the
                             original HbbTV 2.0.1 specification text said to put here.
                             Hence this fixed schema allows it in either place.
                             For best compatibility, applications should look for it in
                             both places.
                            -->
                        <xs:element name="drm" type="termcap:drmType" minOccurs="0"
maxOccurs="unbounded"/>

                        <!-- For future compatibility -->
                        <xs:any namespace="##targetNamespace" processContents="lax"/>
                    </xs:sequence>

                    <!-- For future compatibility -->
                    <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>


        <xs:complexType name="uiProfileType">
            <xs:complexContent>
                <xs:extension base="termcap:uiProfileType">
                    <!-- For future compatibility -->
                    <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>


        <xs:complexType name="uiExtensionType">
            <xs:complexContent>
                <xs:extension base="termcap:uiExtensionType">
                    <xs:choice minOccurs="0" maxOccurs="unbounded">
                        <!-- If adding new elements to this list, please note:

                              * Do NOT add minOccurs/maxOccurs to any entry
                                here.  The parent <choice> element has
                                minOccurs="0" maxOccurs="unbounded" and they
                                will apply to all elements listed here.
```

```
                            Specifying a minOccurs/maxOccurs with those
                            values here will just confuse XML parsers.
                            Specifying different values will have no effect
                            except to confuse humans.
                        -->

                        <!-- Valid values for HbbTV 1.5 / OIPF 1.3 -->
                        <xs:element name="clientMetadata" type="termcap:metadataType"/>
                        <xs:element name="communicationServices" type="xs:boolean"/>
                        <xs:element name="configurationChanges" type="xs:boolean"/>
                        <xs:element name="drm" type="termcap:drmType"/>
                        <xs:element name="extendedAVControl" type="xs:boolean"/>
                        <xs:element name="mdtf" type="xs:boolean"/>
                        <xs:element name="overlayIPbroadcast" type="termcap:overlayType"/>
                        <xs:element name="overlaylocaltuner" type="termcap:overlayType"/>
                        <xs:element name="parentalcontrol" type="termcap:parentalControlType"/>
                        <xs:element name="pollingNotifications" type="xs:boolean"/>
                        <xs:element name="presenceMessaging" type="xs:boolean"/>
                        <xs:element name="recording" type="termcap:pvrType"/>
                        <xs:element name="remote_diagnostics" type="xs:boolean"/>
                        <xs:element name="video_broadcast" type="termcap:videoBroadcastType"/>

                        <!-- HbbTV 2.0.0 was withdrawn, so elements are listed
                             under HbbTV 2.0.1 instead -->

                        <!-- Introduced in OIPF 2.3 (part of HbbTV 2.0.1) -->
                        <xs:element name="hibernateMode" type="xs:boolean"/>
                        <xs:element name="html5_media" type="xs:boolean"/>
                        <xs:element name="playbackControl" type="termcap:playbackType"/>
                        <xs:element name="remoteControlFunction" type="xs:boolean"/>
                        <xs:element name="telephony_services" type="termcap:telephonyServicesType"/>
                        <xs:element name="temporalClipping" type="termcap:hasCapability"/>
                        <xs:element name="wakeupApplication" type="xs:boolean"/>
                        <xs:element name="wakeupOITF" type="xs:boolean"/>
                        <xs:element name="widgets" type="xs:boolean"/>

                        <!-- Introduced in HbbTV 2.0.1 -->
                        <xs:element name="graphicsPerformance"
type="termcap:graphicsPerformanceType"/>

                        <!-- Introduced in HbbTV 2.0.2 -->
                        <xs:element name="broadcast" type="xs:anyURI"/>
                        <xs:element name="video_display_format"
type="termcap:videoDisplayFormatType"/>

                        <!-- Introduced in HbbTV 2.0.3 -->
                        <xs:element name="display_size" type="termcap:displaySizeType"/>
                        <xs:element name="audio_system" type="termcap:audioSystemType"/>
                        <xs:element name="html5_media_variable_rate"
                        type="termcap:html5MediaVariableRateType"/>

                        <!-- For future compatibility and 3rd party extensions -->
                        <xs:any namespace="##any" processContents="lax"/>
                    </xs:choice>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>


        <xs:complexType name="fontFormatType">
            <xs:simpleContent>
                <xs:extension base="termcap:fontFormatType">
                    <!-- For future compatibility -->
                    <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>


        <xs:complexType name="securityType">
            <xs:simpleContent>
                <xs:extension base="termcap:securityType">
                    <!-- For future compatibility -->
                    <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
```

```
        <xs:complexType name="downloadType">
            <xs:simpleContent>
                <xs:extension base="termcap:downloadType">
                    <xs:attribute name="manageDownloads" type="termcap:manageDownloadsType"
default="none"/>

                    <!-- For future compatibility -->
                    <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>


        <xs:complexType name="mimeExtensionType">
            <xs:simpleContent>
                <xs:extension base="termcap:mimeExtensionType">
                    <!-- For future compatibility -->
                    <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>


        <xs:complexType name="audioProfileType">
            <xs:complexContent>
                <xs:extension base="termcap:audioProfileType">
                    <xs:attribute name="DRMSystemID" type="xs:string"/>

                    <!-- Introduced in HbbTV 2.0.1 -->
                    <xs:attribute name="sync_tl" type="termcap:sync_tl_type" use="optional"/>

                    <!-- For future compatibility -->
                    <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>


        <xs:complexType name="videoProfileType">
            <xs:complexContent>
                <xs:extension base="termcap:videoProfileType">
                    <xs:attribute name="DRMSystemID" type="xs:string"/>

                    <!-- Introduced in HbbTV 2.0.1 -->
                    <xs:attribute name="sync_tl" type="termcap:sync_tl_type" use="optional"/>
                    <xs:attribute name="hdr" type="xs:anyURI" use="optional"/>

                    <!-- For future compatibility -->
                    <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>


    </xs:redefine>


    <!-- End CEA-with-OIPF-and-HbbTV-mods -->
    <!-- ************************************************************** -->


    <!-- ************************************************************** -->
    <!-- Start OIPF-with-HbbTV-mods -->

    <!-- Note: We don't reference the OIPF schema, we just copy-paste it
         in here.  This avoids having multiple namespaces.
         -->

    <!--ADDED: type definitions for the new elements defined in Section 9.3 of the
        Open IPTV forum Volume 5 Declarative Application Environment Release 2 specification
        -->
    <xs:simpleType name="manageDownloadsType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="none"/>
            <xs:enumeration value="initiator"/>
            <xs:enumeration value="samedomain"/>
            <xs:enumeration value="all"/>
```

```
        </xs:restriction>
    </xs:simpleType>


    <xs:simpleType name="manageRecordingsType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="none"/>
            <xs:enumeration value="initiator"/>
            <xs:enumeration value="samedomain"/>
            <xs:enumeration value="all"/>
        </xs:restriction>
    </xs:simpleType>


    <xs:complexType name="videoBroadcastType">
        <xs:attribute name="type" type="xs:string" use="required"/>
        <xs:attribute name="transport" type="xs:string"/>
        <xs:attribute name="nrstreams" type="xs:unsignedInt" default="1"/>
        <xs:attribute name="scaling" type="termcap:scalingType" default="arbitrary"/>
        <xs:attribute name="minSize" type="xs:unsignedInt" default="0"/>
        <xs:attribute name="postList" type="xs:boolean" default="false"/>

        <!-- Introduced in OIPF 2.3 (part of HbbTV 2.0.1) -->
        <xs:attribute name="networkTimeshift" type="xs:boolean" default="false"/>
        <xs:attribute name="localTimeshift" type="xs:boolean" default="false"/>

        <!-- For future compatibility -->
        <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
    </xs:complexType>


    <xs:complexType name="pvrType">
        <xs:simpleContent>
            <xs:extension base="xs:boolean">
                <xs:attribute name="ipBroadcast" type="xs:boolean" default="false"/>
                <xs:attribute name="manageRecordings" type="termcap:manageRecordingsType"
default="none"/>
                <xs:attribute name="postList" type="xs:boolean" default="false"/>

                <!-- Introduced in OIPF 2.3 (part of HbbTV 2.0.1) -->
                <xs:attribute name="HAS" type="xs:boolean" default="false"/>
                <xs:attribute name="DASH" type="xs:boolean" default="false"/>

                <!-- For future compatibility -->
                <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>


    <xs:complexType name="parentalControlType">
        <xs:simpleContent>
            <xs:extension base="xs:boolean">
                <xs:attribute name="schemes" type="xs:string"/>

                <!-- For future compatibility -->
                <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>


    <xs:complexType name="metadataType">
        <xs:simpleContent>
            <xs:extension base="xs:boolean">
                <xs:attribute name="type" type="xs:string"/>

                <!-- For future compatibility -->
                <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>


    <xs:complexType name="drmType">
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute name="DRMSystemID" type="xs:string" use="required"/>
```

```
            <xs:attribute name="protectionGateways" type="xs:string" default=""/>

            <!-- For future compatibility -->
            <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>


<!-- Introduced in OIPF 2.3 (part of HbbTV 2.0.1) -->
<xs:complexType name="telephonyServicesType">
    <xs:simpleContent>
        <xs:extension base="xs:boolean">
            <xs:attribute name="video" type="xs:boolean" default="false"/>

            <!-- For future compatibility -->
            <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>


<!-- Introduced in OIPF 2.3 (part of HbbTV 2.0.1) -->
<xs:complexType name="playbackType">
    <xs:simpleContent>
        <xs:extension base="xs:boolean">
            <xs:attribute name="type" type="xs:string"/>

            <!-- For future compatibility -->
            <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>


<!-- Introduced in OIPF 2.3 (part of HbbTV 2.0.1) -->
<xs:complexType name="hasCapability">
    <!-- For future compatibility -->
    <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
</xs:complexType>


<!-- End OIPF-with-HbbTV-mods -->
<!-- ************************************************************** -->


<!-- ************************************************************** -->
<!-- Start HbbTV-defined -->


<!-- Introduced in HbbTV 2.0.1 -->
<xs:simpleType name="sync_tl_type">
    <xs:list itemType="termcap:sync_tl_values_type"/>
</xs:simpleType>


<!-- Introduced in HbbTV 2.0.1 -->
<xs:simpleType name="sync_tl_values_type">
    <xs:restriction base="xs:string">
        <xs:enumeration value="pts"/>
        <xs:enumeration value="ct"/>
        <xs:enumeration value="temi"/>
        <xs:enumeration value="dash_pr"/>
    </xs:restriction>
</xs:simpleType>


<!-- Introduced in HbbTV 2.0.1 -->
<xs:complexType name="graphicsPerformanceType">
    <xs:attribute name="level" type="xs:string"/>

    <!-- For future compatibility -->
    <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
</xs:complexType>


<!-- Introduced in HbbTV 2.0.2 -->
<xs:simpleType name="colorimetryListType">
```

```
        <xs:list itemType="termcap:colorimetryType"/>
    </xs:simpleType>


    <!-- Introduced in HbbTV 2.0.2 -->
    <xs:simpleType name="colorimetryType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="bt709"/>
            <xs:enumeration value="bt2020"/>
        </xs:restriction>
    </xs:simpleType>


    <!-- Introduced in HbbTV 2.0.2 -->
    <xs:complexType name="videoDisplayFormatType">
        <xs:attribute name="width" type="xs:integer" use="required"/>
        <xs:attribute name="height" type="xs:integer" use="required"/>
        <xs:attribute name="frame_rate" type="xs:integer" use="required"/>
        <xs:attribute name="bit_depth" type="xs:integer" use="required"/>
        <xs:attribute name="colorimetry" type="termcap:colorimetryListType" use="required"/>

        <!-- For future compatibility -->
        <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
    </xs:complexType>

    <!-- Introduced in HbbTV TA -->
    <xs:complexType name="taType">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
            <xs:element name="profile" type="xs:anyURI"/>
        </xs:sequence>
        <xs:attribute name="version" type="termcap:versionType" use="required"/>
        <xs:attribute name="broadcastTimelineMonitoring" type="xs:boolean" use="required"/>
        <xs:attribute name="GOPIndependentSwitchToBroadcast" type="xs:boolean" use="required"/>

        <!-- For future compatibility -->
        <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
    </xs:complexType>

    <!-- Introduced in HbbTV TA -->
    <xs:simpleType name="versionType">
        <xs:restriction base="xs:string">
            <xs:pattern value="[1-9][0-9]*\.[1-9][0-9]*\.[1-9][0-9]*"/>
        </xs:restriction>
    </xs:simpleType>

    <!-- Introduced in HbbTV ADB2 and modified in later versions of ADB -->
    <xs:complexType name="hdmiType">
        <xs:attribute name="broadbandOverlay" type="xs:boolean" use="required"/>
        <xs:attribute name="monitoringAWMWhilePlayingBroadband" type="xs:boolean" use="required"/>
        <xs:attribute name="monitoringVWMWhilePlayingBroadband" type="xs:boolean" use="required"/>
        <xs:attribute name="scaling" type="xs:boolean" use="required"/>

        <!-- For future compatibility -->
        <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
    </xs:complexType>

    <!-- Introduced in HbbTV 2.0.3 -->
    <xs:simpleType name="measurementType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="built-in"/>
            <xs:enumeration value="hdmi-accurate"/>
            <xs:enumeration value="hdmi-other"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="displaySizeType">
        <xs:attribute name="width" type="xs:integer" use="required"/>
        <xs:attribute name="height" type="xs:integer" use="required"/>
        <xs:attribute name="measurement_type" type="termcap:measurementType" use="required"/>
        <!-- For future compatibility -->
        <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
    </xs:complexType>

    <xs:complexType name="audioSystemType">
        <xs:attribute name="audio_output_format" type="termcap:audioOutputType"
        use="required"/>
        <xs:attribute name="audio_output_interface" type="audioOutputInterfaceType" use="required"/>
        <!—The following attributes are conditional mandatory depending on audio_output_interface --
>
```

```
        <xs:attribute name="hdmi_audio_pcm" type="xs:boolean" use="optional"/>
        <xs:attribute name="audio_system_between_TV_and_STB" type="xs:boolean" use="optional"/>
        <!-- For future compatibility -->
        <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
    </xs:complexType>
    <xs:simpleType name="audioOutputType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="stereo"/>
            <xs:enumeration value="multichannel"/>
            <xs:enumeration value="multichannel-preferred"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="audioOutputInterfaceType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="local"/>
            <xs:enumeration value="hdmi-arc-sac"/>
            <xs:enumeration value="disabled"/>
            <xs:enumeration value="other"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:complexType name="html5MediaVariableRateType">
        <xs:attribute name="min" type="xs:double" use="required"/>
        <xs:attribute name="max" type="xs:double" use="required"/>
        <!-- For future compatibility -->
        <xs:anyAttribute namespace="##targetNamespace" processContents="lax"/>
    </xs:complexType>


    <!-- End HbbTV-defined -->
    <!-- ************************************************************** -->


</xs:schema>
```

A.2.2 Extensions to the LocalSystem class

The `LocalSystem` class shall be extended with the following additional property.

`readonly Date audioVolumeSetOnTerminal`
Shall return the date and time when the audio volume was last set on the terminal while this HDMI input was selected. Setting audio volume on the terminal while other HDMI inputs are selected shall not modify this value. Setting audio volume on the terminal while non-HDMI sevices, applications or content are selected shall not modify this value. This shall include the terminal receiving volume up/down remote control codes from a remote control (both the terminal's own remote control and a STB remote control configured to send the correct remote control codes for the specific model of terminal). If the terminal supports HDMI CEC 2.0 (uncommon at the present time) then this shall include the STB forwarding volume/mute key events to the TV, avoiding the need for the STB remote to be configured to send the correct remote control codes for the terminal). This forwarding mechanism is not supported in HDMI CEC (as defined in HDMI Specification Version 1.4b). This value shall persist across power cycles following the requirements for writing cookies to persistent storage in table 11,"Minimum terminal capabilities", of TS 102 796 [1].

A.3 Modifications, extensions and clarifications to TS 103 736-1 V1.1.1

A.3.1 Introduction

The following modifications, extensions and clarifications shall apply when the present document is used in combination with V1.1.1 of TS 103 736-1 [15]. It is expected that these will be included in a later revision of that document.

A.3.2 Immediate switch

Use of `urn:hbbtv:sync:timeline:immediate` as the timelineSelector shall indicate that a switch shall happen as soon as possible regardless of the switch preparation deadline and in no case longer than D2max. When this is used, the `switchTime` argument shall be ignored and no requirements on accuracy of switching shall apply.

NOTE: This is intended for switching from broadband back to broadcast when something critical happens in the broadcast.

A.3.3 Method signature of switchMediaPresentation

The following additional signature shall be supported for the `switchMediaPresentation` method.

`Promise switchMediaPresentation( Element originalMediaObject, Element newMediaObject, Object optionalArguments)`		
Description	Requests a switch from the media being presented by originalMediaObject to the media to be presented by newMediaObject.	
Arguments	`originalMediaObject`	Either a video/broadcast object or an HTML5 video element.
	`newMediaObject`	Either a video/broadcast object or an HTML5 video element.
	`optionalArguments`	A JavaScript object corresponding to a WebIDL [17] Dictionary with members as defined below.
Returns	`Promise`	The method always returns a Promise and never raises a synchronous exception.

The `optionalArguments` argument shall support the members (key-value pairs) as defined in table A.16.

Table A.16: Dictionary members

Name / key	Value
timelineSelector	Same as for the argument of the same name to `switchMediaPresentation( Element originalMediaObject, String timelineSelector, Boolean timelineSource, Number switchTime, Element newMediaObject, String minimumSwitchPerformanceRequired)` in TS 103 736-1 [15].
timelineSource	
switchTime	
minimumSwitchPerformanceRequired	
permittedWatermarkStates	A watermark state that is identified by one of the strings: `audio-watermark-detected-only`, `verified-video-watermark-detected-only`, `audio-and-verified-video-watermarks-detected`, `audio-and-unverified-video-watermarks-detected`, or a comma-separated list of two or more of these strings.

Calling `switchMediaPresentation( Element originalMediaObject, Element newMediaObject, Object optionalArguments)` for a transition between two objects with the only optional arguments being some or all of the four optional arguments `timelineSelector`, `timelineSource`, `switchTime` and `minimumSwitchPerformanceRequired` shall be identical to calling `switchMediaPresentation( Element originalMediaObject, String timelineSelector, Boolean timelineSource, Number switchTime, Element newMediaObject, String minimumSwitchPerformanceRequired)` with the same values for each of the corresponding arguments.

The `permittedWatermarkStates` key enables an application to define the states of the watermark state machine (see clause 6.3.1) where a fast switch is permitted to happen. Clause 9.6.2 defines how this state or these states are used.

A.4 Modifications, extensions and clarifications to TS 102 796

A.4.1 Introduction

The following modifications, extensions and clarifications shall apply when the present document is used in combination with versions of TS 102 796 [1] that do not include an equivalent modification or where an equivalent modification is not included in errata that are supported by a terminal. It is expected that these will be included in a future revision of that document.

A.4.2 Exit of a broadcast-related application

When a broadcast-related application exits, the broadcast video and audio shall be presented under the control of the terminal (see clause H.2 of the OIPF DAE specification). Specifically;

- If the application that exited had no video/broadcast object or had a video/broadcast object in the unrealized state then the broadcast video and audio were already being presented under the control of the terminal and this shall continue without interruption.
- If the application that exited had a video/broadcast object in the presenting state then control of broadcast video and audio presentation shall continue but shall return to the terminal (see clause H.2 of the OIPF DAE specification [4]). This may result in changes to video presentation (e.g. video scaling, positioning). This may result in changes to audio presentation (e.g. reverting to a default audio track).
- If the application that exited had a video/broadcast object in the stopped state and was presenting video and audio using an HTML5 video element or an A/V control object, presentation of that video and audio shall be stopped and presentation of broadcast video and audio started under the control of the terminal.

Annex B (normative): Electronic attachments

The present document includes an electronic attachment ts_103464v010201p0.zip with the following contents:

- hbbwm.xsd
  This is the normative XML schema file whose text is included informatively in clause 7.1.2 of the present document.
- adb-xml-ait-watermark-example
  This is an example XML AIT including the XML schema extensions defined in clause 7.1.2 of the present document and which validates using the XML schema.
- hbbtv-capabilities-2019-1.xsd
  This is the normative XML schema file whose text is included informatively in clause A.2.1 to the present document.
- example-adb
  This is an example XML capabilities including the XML schema extensions defined in clauses 10.2.2 and A.2.1 of the present document and which validates using the schema.

For convenience, the electronic attachment also includes the following dependencies from other specifications:

- mis_xmlait.xsd from TS 102 809 [6]

- hbbtv_application_descriptor.xsd from TS 102 796 [1]

The following other XML schema dependencies can be found in the electronic attachments for TS 102 034 [5].

- sdns_v1.4r13.xsd
- sdns_v1.5r25b.xsd
- tva_metadata_3-1_v131.xsd
- tva_metadata_3-1_v171.xsd
- tva_mpeg7.xsd
- tva_mpeg7_2008.xsd
- xml.xsd

XML schema dependencies for the OIPF DAE specification [4] can be found at http://www.oipf.tv/docs/OIPF-Schemas_v2_3-2014-01-24.ZIP.

Annex C (informative): Sequence diagrams

C.1 Application discovery in the presence of DVB Service Information

Figure C.1 illustrates the process of application discovery over broadband using in the presence of DVB Service Information. That this is a simplified diagram and some aspects are omitted in the interests of clarity. This diagram is not a substitute for the normative text in the present document.

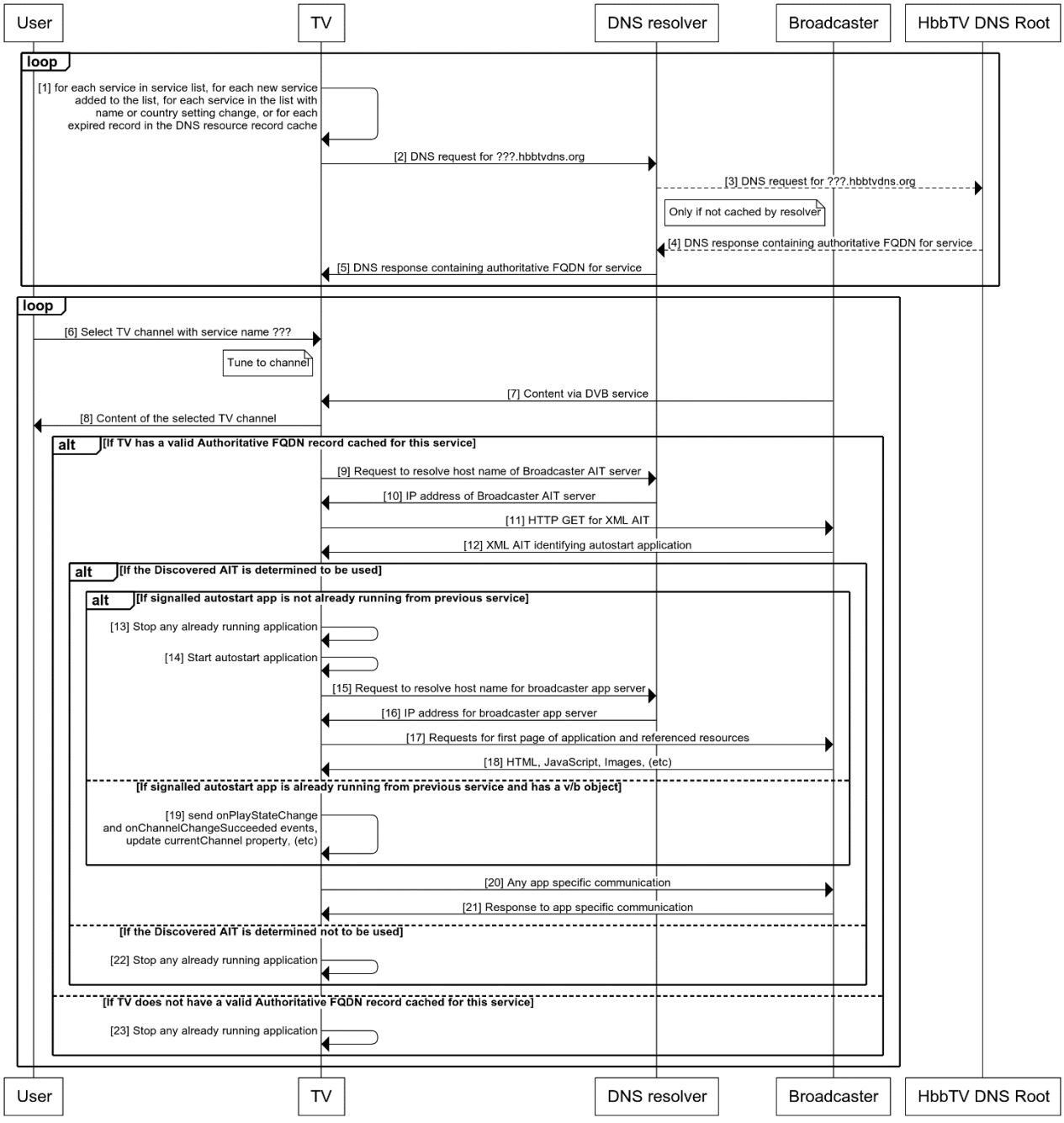

Figure C.1: Sequence diagram for application discovery over broadband in the presence of DVB Service Information

Table C.1 provides an explanation of the individual steps in the diagram. The rows in the table are a 1:1 correspondence with the messages/events in the sequence diagram and are in the same order (top to bottom) as the messages/events in the diagram.

Table C.1: Sequence diagram events/messages for DVB-SI

Step	Between	Label	Comment
	loop		
1	TV->TV	For each service in service list, for each new service added to the list, for each service in the list with name or country setting change, or for each expired record in the DNS resource record cache.	See clause 5.2.
2	TV->DNS resolver	DNS request for ???.hbbtvdns.org.	See clause 5.4.1.
3	DNS resolver-->hbbtvdns.org	DNS request for ???.hbbtvdns.org.	See clause 5.4.1.
4	hbbtvdns.org-->DNS resolver	DNS response containing authoritative FQDN for service.	
5	DNS resolver->TV	DNS response containing authoritative FQDN for service.	
	end of loop		
	loop		
6	User->TV	Select TV channel with service name ???.	Tune to channel.
7	Broadcaster->TV	Content via DVB service.	
8	TV->User	Content of the selected TV channel.	
	alt If TV has a valid DNS resource record for the authoritative FQDN cached for this service.		
9	TV->DNS resolver	Request to resolve host name of Broadcaster AIT server.	According to normal internet specifications.
10	DNS resolver->TV	IP address of Broadcaster AIT server.	
11	TV->Broadcaster	HTTP GET for XML AIT.	See clause 5.6.
12	Broadcaster->TV	XML AIT identifying autostart application.	
	alt If the Discovered AIT is determined to be used.		See clause 6.2.
	alt if signalled autostart app is not already running from previous service		
13	TV->TV	Stop any already running application.	According to regular HbbTV® application lifecycle.
14	TV->TV	Start autostart application.	
15	TV->DNS resolver	Request to resolve host name for broadcaster app server.	According to normal internet specifications.
16	DNS resolver->TV	IP address for broadcaster app server.	
17	TV->Broadcaster	Requests for first page of application and referenced resources.	As for any HbbTV® application.
18	Broadcaster->TV	HTML, JavaScript, Images, etc.	
	else If signalled autostart app is already running from previous service and has a v/b object.		
19	TV->TV	send onPlayStateChange and onChannelChangeSucceeded events, update currentChannel property, etc.	As for regular HbbTV® broadcast-related applications that continue to run after a channel change.
	end of "if signalled autostart app is not already running from previous service".		
20	TV->Broadcaster	Any app specific communication.	Same as for a regular HbbTV® application.
21	Broadcaster->TV	Response to app specific communication.	
	else if the Discovered AIT is determined not to be used.		
22	TV->TV	Stop any already running application.	
	else If TV does not have a valid DNS resource record for the authoritative FQDN cached for this service.		
23	TV->TV	Stop any already running application.	Same as for a regular HbbTV® broadcast-related application when the channel is changed to one with no AIT.
	End of loop.		

C.2 Application discovery using ATSC 3 watermarks

Figure C.2 illustrates the process of application discovery over broadband using ATSC3 watermarks. That this is a simplified diagram and some aspects are omitted in the interests of clarity - e.g. stopping applications when changing to a broadcast that does not contain a watermark. This diagram is not a substitute for the normative text in the specification.

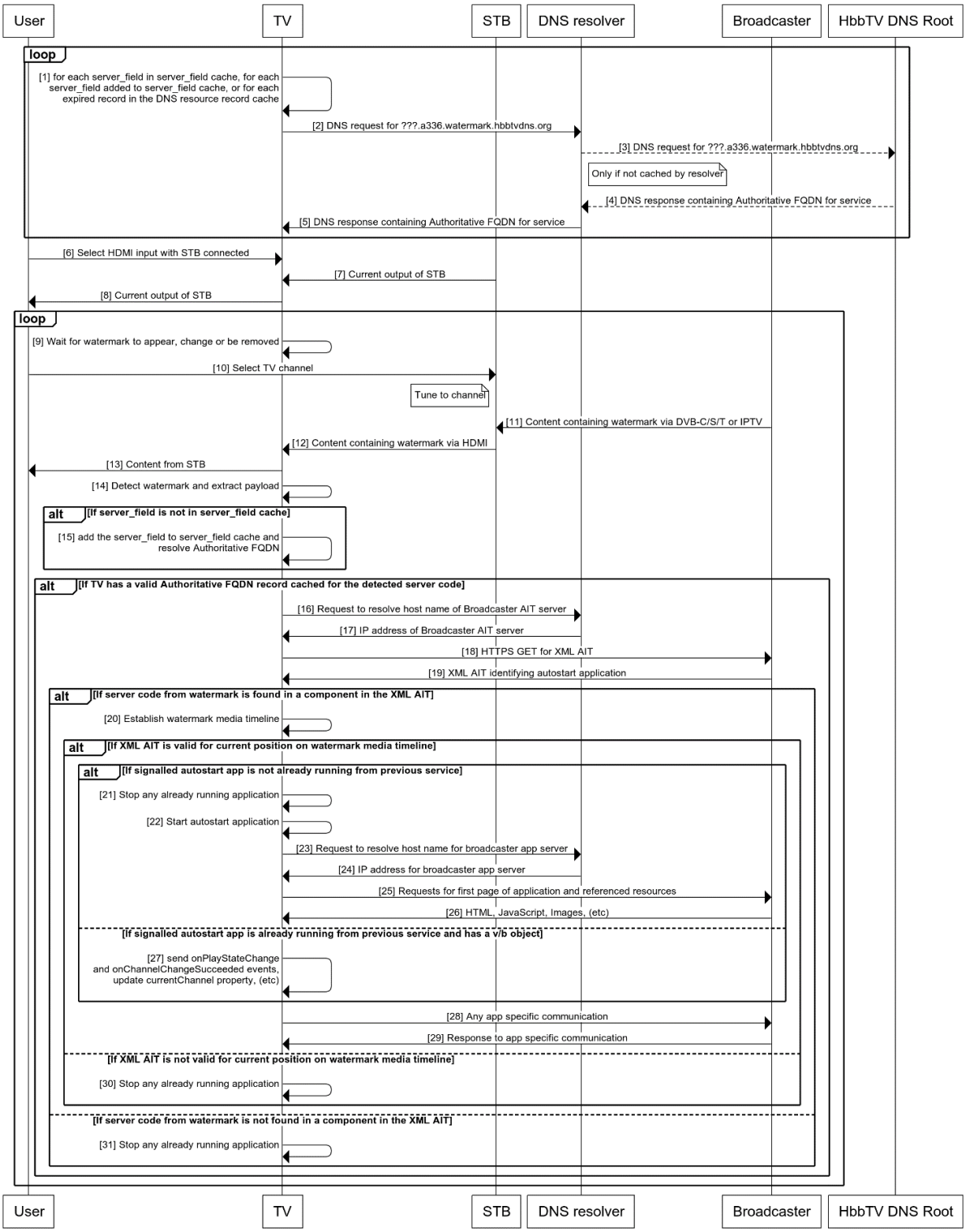

Figure C.2: Sequence diagram for application discovery over broadband using ATSC3 watermarks

Table C.2 provides an explanation of the individual steps in the diagram. The rows in the table are a 1:1 correspondence with the messages/events in the sequence diagram and are in the same order (top to bottom) as the messages/events in the diagram.

Table C.2: Sequence diagram events/messages for watermarking

STEP	Between	Label in sequence diagram	Comment
	loop		
1	TV->TV	For each server_field in server_field cache, each server_field added to server_field cache, or for each expired record in the DNS resource record cache.	See clause 5.2.
2	TV->DNS resolver	DNS request for ???.a336.watermark.hbbtvdns.org.	See clause 5.4.2.
3	DNS resolver-->hbbtvdns.org	DNS request for ???.a336.watermark.hbbtvdns.org.	See clause 5.4.2.
4	hbbtvdns.org-->DNS resolver	DNS response containing authoritative FQDN for service.	
5	DNS resolver->TV	DNS response containing authoritative FQDN for service.	
	end loop		
6	User->TV	Select HDMI input with STB connected.	Application discovery over broadband using watermarks may only be operational when an HDMI input is selected on the TV.
7	STB->TV	Current output of STB.	
8	TV->User	Current output of STB.	
	loop		
9	TV->TV	Wait for watermark to appear, change or be removed.	On a TV supporting the watermark option of the present document , monitoring for watermarks would be active whenever an HDMI input is selected.
10	User->STB	Select TV channel.	
		Tune to channel.	
11	Broadcaster->STB	Content containing watermark via DVB-C/S/T or IPTV.	
12	STB->TV	Content containing watermark via HDMI.	
13	TV->User	Content from STB.	
14	TV->TV	Detect watermark and extract payload.	
	alt If server_field is not in server_field cache.		
15	TV->TV	Add the server_field to server_field cache and resolve Authoritative FQDN.	
	end		
	alt If TV has a valid DNS resource record for the authoritative FQDN cached for the detected server code.		
16	TV->DNS resolver	Request to resolve host name of Broadcaster AIT server.	According to normal internet specifications.
17	DNS resolver->TV	IP address of Broadcaster AIT server.	
18	TV->Broadcaster	HTTP GET for XML AIT.	See clause 5.6.
19	Broadcaster->TV	XML AIT identifying autostart application.	
	alt If server code from watermark is found in a component in the XML AIT.		
20	TV->TV	Establish watermark media timeline.	See clause 6.4.2.4.
	alt If XML AIT is valid for current position on watermark media timeline.		
	alt If signalled autostart app is not already running from previous service.		According to regular HbbTV® application lifecycle.
21	TV->TV	Stop any already running application.	
22	TV->TV	Start autostart application.	
23	TV->DNS resolver	Request to resolve host name for broadcaster app server.	As for any HbbTV® application delivered over broadband.
24	DNS resolver->TV	IP address for broadcaster app server.	

STEP	Between	Label in sequence diagram	Comment
25	TV->Broadcaster	Requests for first page of application and referenced resources.	
26	Broadcaster->TV	HTML, JavaScript, Images, etc.	
	else If signalled autostart app is already running from previous service and has a v/b object.		
27	TV->TV	Send onPlayStateChange\and onChannelChangeSucceeded events, update currentChannel property, etc.	As for regular HbbTV® broadcast-related applications that continue to run after a channel change.
	end of " If signalled autostart app is not already running from previous service".		
28	TV->Broadcaster	Any app specific communication.	Same as for a regular HbbTV® application.
29	Broadcaster->TV	Response to app specific communication.	
	else If XML AIT is not valid for current position on watermark media timeline.		
30	TV->TV	Stop any already running application.	Same as for a regular HbbTV® broadcast-related application when the channel is changed to one with no AIT.
	end of "If XML AIT is not valid for current position on watermark media timeline".		
	else If server code from watermark is not found in a component in the XML AIT.		
31	TV->TV	Stop any already running application.	Same as for a regular HbbTV® broadcast-related application when the channel is changed to one with no AIT.
	end of "If server code from watermark is not found in a component in the XML AIT".		
	end of "TV has a valid Authoritative FQDN record cached for the detected server code".		

Annex D (informative): Targeted advertising using watermarks over HDMI

D.1 HDMI Audio configuration

It is important that the user have a good experience when switching from broadcast to an advert (or other replacement content) and back again. This is particularly challenging for audio, for example;

- The audio for the advert needs to come out of the same speakers as the audio for the broadcast
- The audio for the advert needs to have the same volume as the audio for the broadcast. This may be a legal or regulatory requirement.

These requirements may not be possible to meet depending on how individual users have configured the audio between the STB, the TV and any optional audio amplifier or sound bar.

The present document defines how applications are able to obtain information about the audio configuration that can be used by the application to decide if the audio configuration is one where it would be safe to perform advert replacement. The present document does not define requirements on how an application makes this decision but one example of a possible decision process is in figure D.1 below.

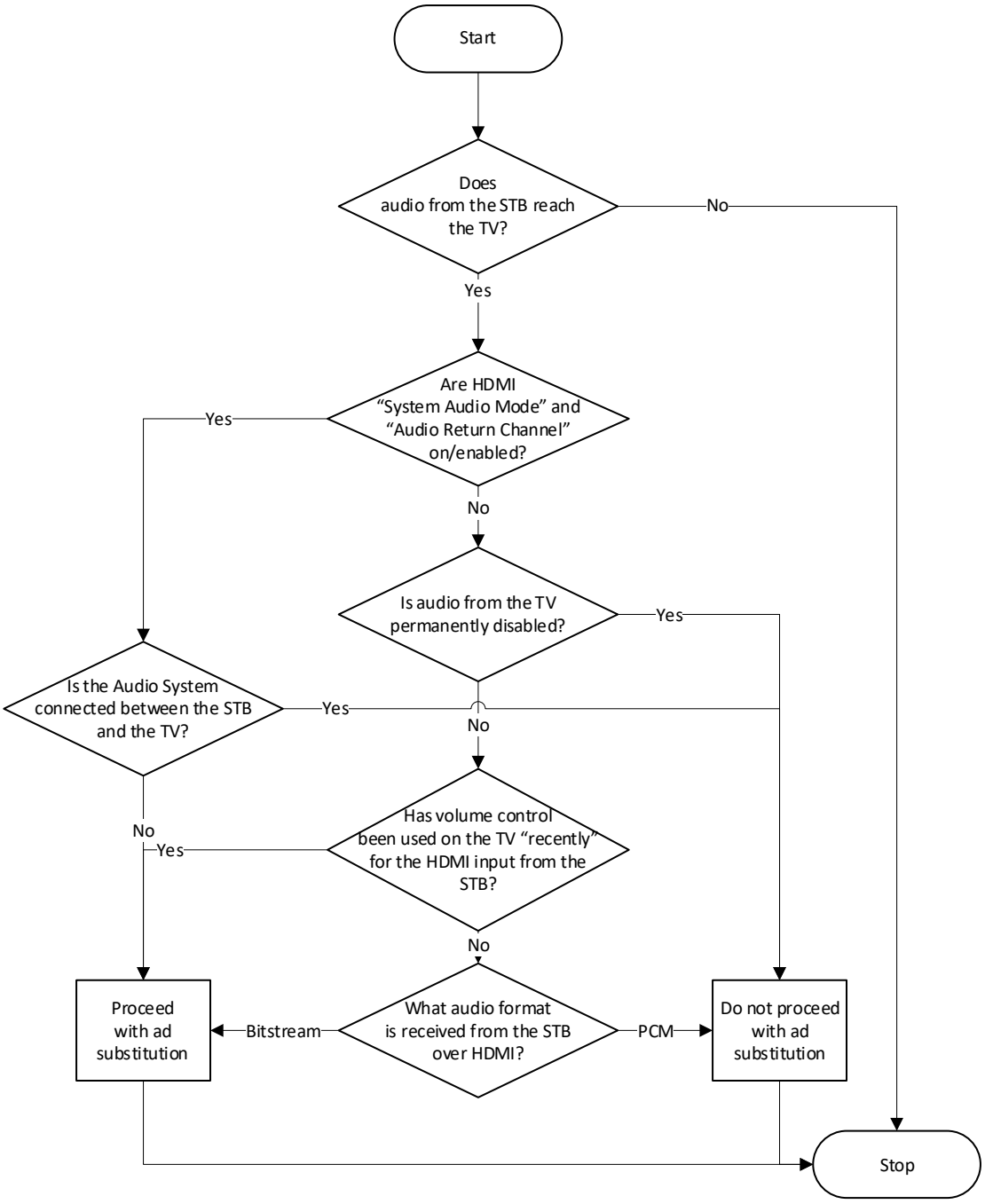

Figure D.1:Example audio configuration decision process

Some details of the steps in this example decision process are as follows;

- **Does audio from the STB reach the TV?** If the audio does not reach the TV then the DAS application will never be started.
- **Are HDMI "System Audio Mode" and "Audio Return Channel" on/enabled?** If the TV speakers are disabled and HDMI "system audio mode" is enabled and HDMI audio return channel (or enhanced audio return channel) is enabled then it can be assumed that an audio system (e.g. Home Theatre System, soundbar, …) is being used and that this external audio system can be controlled by the TV. Applications can query this using the `audio-output-interface` attribute in the XML capabilities defined in clause 10.2.2.
- **Is the audio system connected between the TV and the STB?** If both the STB and the TV are both directly connected to the external audio system then, if the TV would switch the external audio system from the STB to the TV then this might interrupt the audio from the STB reaching the TV. If this would happen then the DAS

application would be killed. Applications can query this using the attribute `audio_system_between_TV_and_STB` in the XML capabilities defined in clause 10.2.2.

- **Is audio from the TV permanently disabled?** This corresponds to 3 different possible configurations of the TV by the user. In all cases, this can be taken as an indication that some kind of audio system is in use that does not support HDMI "system audio mode" and "audio return channel". Applications can query this using the `audio-output-interface` attribute in the XML capabilities defined in clause 10.2.2.
  - The user has explicitly disabled audio on a TV that has a UI option to disable speakers or similar.
  - The user has implicitly disabled audio by permanently setting the volume to zero
  - The user has implicitly disabled audio by permanently muting the audio
- **Has volume control been used on the TV "recently" for the HDMI input from the STB?** If the TV is modifying the volume of the audio received from the STB over HDMI then it can be assumed that audio decoded locally on the TV (e.g. from an advert downloaded over broadband) will be output at the same volume as the audio received from the STB over HDMI. Applications can query this reading the `audioVolumeSetOnTerminal` property.
- **What is the audio format on HDMI?** In the absence of an external audio system, the volume of the audio passed from the STB to the TV can either be modified on the TV or on the STB. If the volume is modified on the STB then audio decoded locally on the TV (e.g. from an advert downloaded over broadband) will be output at a different volume from the audio received from the STB over HDMI. In HDMI, audio may be transferred either in a PCM format or in a compressed format. If the audio volume is being modified by the STB then use of the PCM format is much more likely than use of the compressed format. Applications can query this using the attribute `hdmi_audio_pcm` in the XML capabilities defined in clause 10.2.2.
- **Proceed with ad substitution;** In this example decision process, it is safe for the DAS application to proceed with an advert replacement if either;
  - The audio from the STB has its volume modified on the TV before being output by the TV using local speakers or speakers or an external audio system connected using interfaces such as S/PDIF or Bluetooth; or
  - The audio from the STB is output by an HDMI (e)ARC audio system connected to the TV and where the audio from the STB passes through the TV on the way to the audio system.
- **Do not proceed with ad substitution**; In this example decision process, it is not safe for the DAS application to proceed with an advert replacement if either;
  - The audio from the STB has its volume modified on the STB before being passed to the TV or to an audio system; or
  - The audio from the STB is output by an HDMI (e)ARC audio system connected between the TV and the STB; or
  - The audio configuration is something else that either cannot be identified or where it may be implementation specific whether the same audio volume would be applied to audio from the STB or audio locally decoded on the TV.

D.2 Operations performed by upstream of a terminal

Tables D.2 and D.3 summarise the desired user experience, required behaviour by terminals (based on requirements elsewhere in the present document) and the desired behaviour by applications when a variety of operations are performed upstream of a terminal (e.g. on a STB connected to a terminal via HDMI).

Table D.2: Operation performed while switch is pending

Content Modification	Desired user experience	Required Terminal behaviour	Application behaviour
Channel Change	Advert is not presented.	TA is aborted - see clause 8.1.4.2 of TS 103 736-1 [1], “If originalMediaObject is a video/broadcast object and a channel change succeeds”.	No special behaviour required.
Pause	Advert is not presented.	When both video and audio watermarks are used then 1) a "WatermarkStateChange" event will be posted on the `oipfApplicationManager` reporting transition from ‘Audio and verified video watermarks detected’ to ‘Verified video watermark detected only’ (4) and 2) a `ratechange` event wil be posted on the `MediaSynchroniser` object. If only audio watermarks are used, see (2).	Applications should register for `ratechange` events and abort advert presentation when appropriate. The latter may be done by changing the src attribute of the HTML5 video element.
Speed change ("trick play")	Either advert is not presented or advert is presented synchronized to media time in STB output (1).	If video watermarks are in use then `ratechange` and `timeupdate` events will be posted on the `MediaSynchroniser` object. If only audio watermarks are in use then see (2).	Applications should register for `ratechange` and `timeupdate` events and may abort advert presentation when appropriate. The latter may be done by changing the src attribute of the HTML5 video element.
Skip forward / back ("cue")	Either advert is not presented or advert is presented synchronized to media time in STB output (1).	If both video and audio watermarks are used then "WatermarkStateChange" events will be posted on the `oipfApplicationManager`. reporting transition from “Audio and verified video watermarks detected” to “verified video watermark only” and back again. If only audio watermarks are used then "WatermarkStateChange" events will be posted on the `oipfApplicationManager`. reporting transition from “audio watermark only detected” to “no watermark detected” and back again. The loss of watermark procedure will be followed. For each skip, the watermark media timeline is re-initialized according to clause 6.4.2.4.2, the application lifecycle rules followed and a `timeupdate` event posted on the `MediaSynchroniser` object.	Applications should register for `timeupdate` events and may abort advert presentation when appropriate. The latter may be done by changing the src attribute of the HTML5 video element.
Full-screen UI (e.g. EPG)	Either advert is not presented or advert is presented with audio only for duration of STB action	If both video and audio watermarks are used then a "WatermarkStateChange" event will be posted on the `oipfApplicationManager` reporting transition from ‘Audio and verified video watermarks detected’ to ‘Audio watermark detected only’. If only audio watermarks are used then nothing (3).	Applications should register for `WatermarkStateChange` and either abort advert presentation when appropriate (by changing the src attribute of the HTML5 video element) or set `VideoTrack.selected` to false but let the audio continue.

Mute	Either advert is not presented or advert is presented with video only for duration of STB action	If the audio is muted on the STB then; - If both video and audio watermarks are used then a "WatermarkStateChange" event will be posted on the `oipfApplicationManager` reporting transition from 'Audio and verified video watermarks detected' to 'Verified video watermark detected only'. - If only audio watermarks are used then see (2). If the audio is delivered to the TV and mute is applied in the TV then `LocalSystem.mute` is set to true.	Applications should register for `WatermarkStateChange` events and abort advert presentation when appropriate. The latter may be done by changing the src attribute of the HTML5 video element. Applications may poll `LocalSystem.mute` to discover if the audio mute is applied in the TV.

NOTE 1: The present document does not explicitly support presenting adverts synchronized to media time in the STB output where that media time either has discontinuities or has a playbackRate other than 1.0.

NOTE 2: If only the audio watermark is being used and it is lost then this will result in a transition from 'Audio watermark detected only' to 'No watermark detected'. The loss of watermark process will be run resulting in the application being stopped or hidden as defined in clause 6.4.3. If the application is hidden then the pending switch will be aborted.

NOTE 3: If only the audio watermark is being used and a full screen UI shown on the STB but the broadcast audio kept running by the STB then the TV will not be able to detect this.

NOTE 4: This table assumes that, when the video watermark is present, it is present in all frames of video. Hence when the video watermark is in use, loss of the audio watermark causes a transition from 'Audio and verified video watermarks detected' to 'Verified video watermark detected only' and does not result in 2 transitions ending up at 'No watermark detected'. If the video watermark would only be present in some frames then there may be some circumstances when both watermarks are lost. This would result in 2 `WatermarkStateChange` events, either 'Audio and verified video watermarks detected' to 'Verified video watermark detected only' to 'No watermark detected' or 'Audio and verified video watermarks detected' to 'Audio watermark detected only' to 'No watermark detected'. When both watermarks are lost at essentially the same time, the order in which the events are generated is implementation specific.

Table D.3: Operation performed while switch is in progress

Content Modification	Desired user experience	Required Terminal behaviour	Application behaviour
Channel Change	Advert presentation stops, revert to HDMI.	A new AIT will be retrieved based on any watermark in new channel and the application lifecycle rules obeyed. If the application is stopped then the advert presentation will be stopped and HDMI content shown.	None
Pause	Advert presentation is paused.	If both video and audio watermarks are used then 1) a "WatermarkStateChange" event will be posted on the `oipfApplicationManager` reporting transition from 'Audio and verified video watermarks detected' to 'Verified video watermark detected only' and 2) A `ratechange` event will be posted on the `MediaSynchroniser` object reporting a playbackRate of zero. If only the audio watermark is being used then a pause will result in the 'loss of watermark' process being run and the application being either stopped or hidden as defined in clause 6.4.3.	Applications should register for `ratechange` events and pause advert presentation when appropriate.
Speed change ("trick play")	Present TA synchronized to media time in STB output	Same as pause except that a `ratechange` event and a number of `timeupdate` events will be posted on the `MediaSynchroniser` object. The `ratechange` event will report either an accurate value or NaN. If the `ratechange` event reports an accurate value then the `timeupdate` events will have the `reason` argument being "linear". If the `ratechange` event reports NaN then the `timeupdate` events will have the `reason` argument being "other".	Applications should register for `timeupdate` and `ratechange` events and may change the `playbackRate` for advert presentation or abort advert presentation (see clause A.3.2) as appropriate.
Skip forward / back ("cue")	Present TA synchronized to media time in STB output	The watermark media timeline will be re-initialized according to clause 6.4.2.4.2. If needed, the Discovered AIT will be updated for the new value of watermark media timeline and the application lifecycle rules obeyed. A `timeupdate` event will be posted on the `MediaSynchroniser` object with the `reason` argument being "other' and the `currentTime` argument reporting the media time after the skip..	Applications should register for `timeupdate` events and may seek forwards or backwards within the advert by modifying `currentTime` or abort advert presentation (see clause A.3.2) when appropriate.
Full-screen UI (e.g. EPG)	Present TA with audio only for duration of STB action	If both video and audio watermarks are used then a "WatermarkStateChange" event will be posted on the `oipfApplicationManager` reporting transition from 'Audio and verified video watermarks detected' to 'Audio watermark detected only'. If only the audio watermark is being used then showing a full screen UI on the STB while keeping broadcast audio running is not detected by the TV.	Applications should register for `WatermarkStateChange` events and supress presentation of the video of the advert when appropriate. The latter may be done by setting `VideoTrack.selected` to false.

Mute	Present TA with audio muted for duration of STB action	If the audio is muted on the STB then; - If both video and audio watermarks are used then a "WatermarkStateChange" event will be posted on the `oipfApplicationManager` reporting transition from 'Audio and verified video watermarks detected' to 'Verified video watermark detected only'. - If only audio watermarks are used then mute will result in the 'loss of watermark' process being run and the application being either stopped or hidden as defined in clause 6.4.3. If the app is stopped then HDMI video will be presented automatically by the terminal. If the audio is delivered to the TV and mute is applied in the TV then `LocalSystem.mute` is set to true.	Applications should register for `WatermarkStateChange` events and supress presentation of the audio of the advert when appropriate by setting `AudioTrack.enabled` to false. Applications may poll `LocalSystem.mute` to discover if the audio mute is applied in the TV.

History

Document history		
V1.1.1	September 2016	Publication
V1.2.1	May 2020	Publication