



BSI Standards Publication

## Consumer terminal function for access to IPTV and open multimedia services

---

Part 5-1: Declarative application environment

## National foreword

This British Standard is the UK implementation of IEC 62766-5-1:2017.

The UK participation in its preparation was entrusted to Technical Committee EPL/100, Audio, video and multimedia systems and equipment.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

© The British Standards Institution 2017  
Published by BSI Standards Limited 2017

ISBN 978 0 580 88961 5

ICS 33.170; 35.240.95

**Compliance with a British Standard cannot confer immunity from legal obligations.**

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 31 October 2017.

### Amendments/corrigenda issued since publication

| Date | Text affected |
|------|---------------|
|------|---------------|

---

# INTERNATIONAL STANDARD



---

**Consumer terminal function for access to IPTV and open internet multimedia services –  
Part 5-1: Declarative application environment**

INTERNATIONAL  
ELECTROTECHNICAL  
COMMISSION

ICS 33.170 35.240.95

ISBN 978-2-8322-4573-6

**Warning! Make sure that you obtained this publication from an authorized distributor.**

## CONTENTS

|                                                                                 |    |
|---------------------------------------------------------------------------------|----|
| FOREWORD.....                                                                   | 12 |
| INTRODUCTION.....                                                               | 14 |
| 1 Scope.....                                                                    | 15 |
| 2 Normative references .....                                                    | 15 |
| 3 Terms, definitions and abbreviated terms .....                                | 17 |
| 3.1 Terms and definitions.....                                                  | 17 |
| 3.2 Abbreviated terms.....                                                      | 19 |
| 4 DAE overview .....                                                            | 19 |
| 4.1 General.....                                                                | 19 |
| 4.2 Architecture of the DAE .....                                               | 20 |
| 4.3 Gateway discovery and control .....                                         | 21 |
| 4.4 Application definition.....                                                 | 22 |
| 4.4.1 General .....                                                             | 22 |
| 4.4.2 Similarities between applications and traditional web pages .....         | 22 |
| 4.4.3 Differences between applications and traditional web pages.....           | 22 |
| 4.4.4 The application tree .....                                                | 23 |
| 4.4.5 The application display model.....                                        | 23 |
| 4.4.6 The security model .....                                                  | 23 |
| 4.4.7 Inheritance of permissions .....                                          | 24 |
| 4.4.8 Privileged application APIs .....                                         | 24 |
| 4.4.9 Active applications list .....                                            | 24 |
| 4.4.10 Widgets .....                                                            | 24 |
| 4.4.11 Origin for broadcast-delivered documents.....                            | 25 |
| 4.5 Resource management .....                                                   | 25 |
| 4.5.1 General .....                                                             | 25 |
| 4.5.2 Application lifecycle issues .....                                        | 25 |
| 4.5.3 Caching of application files .....                                        | 26 |
| 4.5.4 Memory usage .....                                                        | 26 |
| 4.5.5 Instantiating embedded objects and claiming scarce system resources ..... | 26 |
| 4.5.6 Media control.....                                                        | 26 |
| 4.5.7 Use of the display .....                                                  | 27 |
| 4.5.8 Cross-application event handling .....                                    | 28 |
| 4.5.9 Tuner resources .....                                                     | 29 |
| 4.6 Parental access control.....                                                | 30 |
| 4.7 Content download .....                                                      | 31 |
| 4.7.1 General .....                                                             | 31 |
| 4.7.2 Download manager.....                                                     | 31 |
| 4.7.3 Content access download descriptor.....                                   | 31 |
| 4.7.4 Triggering a download .....                                               | 31 |
| 4.7.5 Download protocol(s).....                                                 | 32 |
| 4.8 Streaming CoD .....                                                         | 33 |
| 4.8.1 General .....                                                             | 33 |
| 4.8.2 Unicast streaming .....                                                   | 33 |
| 4.9 Scheduled content .....                                                     | 34 |
| 4.9.1 General .....                                                             | 34 |
| 4.9.2 Conveyance of channel list .....                                          | 34 |
| 4.9.3 Conveyance of channel list and list of scheduled recordings.....          | 35 |

|        |                                                              |    |
|--------|--------------------------------------------------------------|----|
| 4.10   | DLNA RUI remote control function .....                       | 36 |
| 4.10.1 | General .....                                                | 36 |
| 4.10.2 | Interfaces used by the DLNA RUI remote control function..... | 37 |
| 4.11   | Power consumption.....                                       | 38 |
| 4.11.1 | General .....                                                | 38 |
| 4.11.2 | DAE application wake-up support .....                        | 39 |
| 4.11.3 | OITF hibernate support.....                                  | 40 |
| 4.11.4 | State diagram for the power state .....                      | 41 |
| 4.12   | Display model .....                                          | 41 |
| 5      | DAE application model .....                                  | 41 |
| 5.1    | Application lifecycle .....                                  | 41 |
| 5.1.1  | General .....                                                | 41 |
| 5.1.2  | Creating a new application.....                              | 41 |
| 5.1.3  | Stopping an application .....                                | 43 |
| 5.1.4  | Application boundaries .....                                 | 43 |
| 5.2    | Application announcement and signalling.....                 | 43 |
| 5.2.1  | Overview .....                                               | 43 |
| 5.2.2  | General .....                                                | 44 |
| 5.2.3  | Broadcast-related applications.....                          | 45 |
| 5.2.4  | Service provider related applications .....                  | 50 |
| 5.2.5  | Broadcast-independent applications .....                     | 51 |
| 5.2.6  | Switching between applications .....                         | 51 |
| 5.2.7  | Signalling format.....                                       | 51 |
| 5.2.8  | Widgets lifecycle.....                                       | 55 |
| 5.3    | Event notifications .....                                    | 56 |
| 5.3.1  | General .....                                                | 56 |
| 5.3.2  | Event notification framework based on CEA 2014 .....         | 57 |
| 5.3.3  | IMS event notification framework .....                       | 59 |
| 6      | Formats .....                                                | 66 |
| 6.1    | Web standards TV profile.....                                | 66 |
| 6.1.1  | General .....                                                | 66 |
| 6.1.2  | Additional restrictions and requirements .....               | 67 |
| 6.2    | Still image formats .....                                    | 67 |
| 6.3    | Media formats .....                                          | 67 |
| 6.3.1  | General .....                                                | 67 |
| 6.3.2  | Media format of A/V media except for audio from memory ..... | 67 |
| 6.3.3  | Media format of A/V media for audio from memory.....         | 68 |
| 6.3.4  | Media transport .....                                        | 68 |
| 6.4    | SVG.....                                                     | 68 |
| 7      | APIs .....                                                   | 68 |
| 7.1    | Object factory API.....                                      | 68 |
| 7.1.1  | General .....                                                | 68 |
| 7.1.2  | Methods .....                                                | 69 |
| 7.1.3  | Examples.....                                                | 71 |
| 7.2    | Application management APIs.....                             | 72 |
| 7.2.1  | General .....                                                | 72 |
| 7.2.2  | The application/oipfApplicationManager embedded object ..... | 72 |
| 7.2.3  | Application class.....                                       | 76 |
| 7.2.4  | The ApplicationCollection class .....                        | 79 |

|        |                                                                                               |     |
|--------|-----------------------------------------------------------------------------------------------|-----|
| 7.2.5  | The ApplicationPrivateData class .....                                                        | 79  |
| 7.2.6  | The Keyset class .....                                                                        | 80  |
| 7.2.7  | New DOM events for application support .....                                                  | 82  |
| 7.2.8  | Examples.....                                                                                 | 83  |
| 7.2.9  | Widget APIs.....                                                                              | 84  |
| 7.3    | Configuration and setting APIs.....                                                           | 85  |
| 7.3.1  | General .....                                                                                 | 85  |
| 7.3.2  | The application/oipfConfiguration embedded object .....                                       | 85  |
| 7.3.3  | The Configuration class .....                                                                 | 86  |
| 7.3.4  | The LocalSystem class .....                                                                   | 88  |
| 7.3.5  | The NetworkInterface class .....                                                              | 94  |
| 7.3.6  | The AVOutput class .....                                                                      | 94  |
| 7.3.7  | The NetworkInterfaceCollection class .....                                                    | 98  |
| 7.3.8  | The AVOutputCollection class .....                                                            | 98  |
| 7.3.9  | The TunerCollection class .....                                                               | 98  |
| 7.3.10 | The Tuner class.....                                                                          | 98  |
| 7.3.11 | The SignalInfo class .....                                                                    | 99  |
| 7.3.12 | The LNBInfo class .....                                                                       | 100 |
| 7.3.13 | The StartupInformation class .....                                                            | 101 |
| 7.4    | Content download APIs.....                                                                    | 101 |
| 7.4.1  | General .....                                                                                 | 101 |
| 7.4.2  | The application/oipfDownloadTrigger embedded object .....                                     | 101 |
| 7.4.3  | Extensions to application/oipfDownloadTrigger .....                                           | 104 |
| 7.4.4  | The application/oipfDownloadManager embedded object.....                                      | 104 |
| 7.4.5  | The Download class .....                                                                      | 110 |
| 7.4.6  | The DownloadCollection class .....                                                            | 113 |
| 7.4.7  | The DRMControllInformation class .....                                                        | 113 |
| 7.4.8  | The DRMControllInfoCollection class.....                                                      | 114 |
| 7.5    | Content on demand metadata APIs .....                                                         | 114 |
| 7.5.1  | General .....                                                                                 | 114 |
| 7.5.2  | The application/oipfCodManager embedded object .....                                          | 114 |
| 7.5.3  | The ContentCatalogueCollection class.....                                                     | 116 |
| 7.5.4  | The ContentCatalogue class.....                                                               | 116 |
| 7.5.5  | The ContentCatalogueEvent class .....                                                         | 117 |
| 7.5.6  | The CODFolder class .....                                                                     | 117 |
| 7.5.7  | The CODAsset class.....                                                                       | 118 |
| 7.5.8  | The CODService class.....                                                                     | 121 |
| 7.6    | Content service protection API.....                                                           | 123 |
| 7.6.1  | General .....                                                                                 | 123 |
| 7.6.2  | The application/oipfDrmAgent embedded object .....                                            | 123 |
| 7.7    | Gateway discovery and control APIs .....                                                      | 125 |
| 7.7.1  | General .....                                                                                 | 125 |
| 7.7.2  | The application/oipfGatewayInfo embedded object .....                                         | 126 |
| 7.8    | Communication services APIs.....                                                              | 128 |
| 7.8.1  | General .....                                                                                 | 128 |
| 7.8.2  | The application/oipfCommunicationServices embedded object.....                                | 129 |
| 7.8.3  | Extensions to application/oipfCommunicationServices for presence and messaging services ..... | 132 |
| 7.8.4  | The UserData class – Properties .....                                                         | 135 |

|         |                                                                                           |     |
|---------|-------------------------------------------------------------------------------------------|-----|
| 7.8.5   | The UserDataCollection class .....                                                        | 136 |
| 7.8.6   | The FeatureTag class – Properties .....                                                   | 136 |
| 7.8.7   | The FeatureTagCollection class .....                                                      | 136 |
| 7.8.8   | The Contact class – Properties .....                                                      | 136 |
| 7.8.9   | The ContactCollection class .....                                                         | 136 |
| 7.8.10  | Extensions to application/oipfCommunicationServices for voice<br>telephony services ..... | 137 |
| 7.8.11  | Extensions to application/oipfCommunicationServices for video<br>telephony services ..... | 143 |
| 7.8.12  | The DeviceInfo class .....                                                                | 145 |
| 7.8.13  | The DeviceInfoCollection class .....                                                      | 146 |
| 7.8.14  | The CodecInfo class .....                                                                 | 146 |
| 7.8.15  | The CodecInfoCollection class .....                                                       | 146 |
| 7.9     | Parental rating and parental control APIs .....                                           | 147 |
| 7.9.1   | General .....                                                                             | 147 |
| 7.9.2   | The application/oipfParentalControlManager embedded object .....                          | 147 |
| 7.9.3   | The ParentalRatingScheme class .....                                                      | 150 |
| 7.9.4   | The ParentalRatingSchemeCollection class .....                                            | 151 |
| 7.9.5   | The ParentalRating class .....                                                            | 152 |
| 7.9.6   | The ParentalRatingCollection class .....                                                  | 154 |
| 7.10    | Scheduled Recording APIs .....                                                            | 155 |
| 7.10.1  | General .....                                                                             | 155 |
| 7.10.2  | The application/oipfRecordingScheduler embedded object .....                              | 155 |
| 7.10.3  | The ScheduledRecording class .....                                                        | 158 |
| 7.10.4  | The ScheduledRecordingCollection class .....                                              | 162 |
| 7.10.5  | Extension to application/oipfRecordingScheduler for control of<br>recordings .....        | 162 |
| 7.10.6  | The Recording class .....                                                                 | 164 |
| 7.10.7  | The RecordingCollection class .....                                                       | 167 |
| 7.10.8  | The PVREvent class .....                                                                  | 167 |
| 7.10.9  | The Bookmark class .....                                                                  | 167 |
| 7.10.10 | The BookmarkCollection class .....                                                        | 168 |
| 7.11    | Remote Management APIs .....                                                              | 168 |
| 7.11.1  | General .....                                                                             | 168 |
| 7.11.2  | The application/oipfRemoteManagement embedded object .....                                | 168 |
| 7.12    | Metadata APIs .....                                                                       | 172 |
| 7.12.1  | General .....                                                                             | 172 |
| 7.12.2  | The application/oipfSearchManager embedded object .....                                   | 173 |
| 7.12.3  | The MetadataSearch class .....                                                            | 175 |
| 7.12.4  | The Query class .....                                                                     | 180 |
| 7.12.5  | The SearchResults class .....                                                             | 181 |
| 7.12.6  | The MetadataSearchEvent class .....                                                       | 182 |
| 7.12.7  | The MetadataUpdateEvent class .....                                                       | 182 |
| 7.13    | Scheduled content and hybrid tuner APIs .....                                             | 182 |
| 7.13.1  | General .....                                                                             | 182 |
| 7.13.2  | The video/broadcast embedded object .....                                                 | 182 |
| 7.13.3  | Extensions to video/broadcast for recording and time-shift .....                          | 198 |
| 7.13.4  | Extensions to video/broadcast for access to EIT p/f .....                                 | 207 |
| 7.13.5  | Extensions to video/broadcast for playback of selected components .....                   | 208 |
| 7.13.6  | Extensions to video/broadcast for parental ratings errors .....                           | 209 |

|         |                                                                                                 |     |
|---------|-------------------------------------------------------------------------------------------------|-----|
| 7.13.7  | Extensions to video/broadcast for DRM rights errors.....                                        | 210 |
| 7.13.8  | Extensions to video/broadcast for current channel information.....                              | 211 |
| 7.13.9  | Extensions to video/broadcast for creating channel lists from SD&S fragments .....              | 211 |
| 7.13.10 | The ChannelConfig class .....                                                                   | 211 |
| 7.13.11 | The ChannelList class .....                                                                     | 216 |
| 7.13.12 | The Channel class .....                                                                         | 217 |
| 7.13.13 | The FavouriteListCollection class .....                                                         | 222 |
| 7.13.14 | The FavouriteList class .....                                                                   | 223 |
| 7.13.15 | Extensions to video/broadcast for channel scan.....                                             | 225 |
| 7.13.16 | The ChannelScanEvent class .....                                                                | 225 |
| 7.13.17 | The ChannelScanOptions class .....                                                              | 225 |
| 7.13.18 | The ChannelScanParameters class .....                                                           | 225 |
| 7.13.19 | The DVBTChannelScanParameters class .....                                                       | 225 |
| 7.13.20 | The DVBSChannelScanParameters class .....                                                       | 227 |
| 7.13.21 | The DVBCChannelScanParameters class.....                                                        | 228 |
| 7.13.22 | Extensions to video/broadcast for synchronization.....                                          | 229 |
| 7.13.23 | The ATSCChannelScanParameters class .....                                                       | 230 |
| 7.14    | Media playback APIs.....                                                                        | 231 |
| 7.14.1  | General .....                                                                                   | 231 |
| 7.14.2  | The A/V Control object.....                                                                     | 231 |
| 7.14.3  | Extensions to A/V Control object for playback through Content-Access Streaming Descriptor ..... | 238 |
| 7.14.4  | Extensions to A/V Control object for trickmodes.....                                            | 239 |
| 7.14.5  | Extensions to A/V Control object for playback of selected components .....                      | 240 |
| 7.14.6  | Extensions to A/V Control object for parental rating errors .....                               | 240 |
| 7.14.7  | Extensions to A/V Control object for DRM rights errors .....                                    | 242 |
| 7.14.8  | Extensions to A/V Control object for playing media objects .....                                | 243 |
| 7.14.9  | Extensions to A/V Control object for UI feedback of buffering A/V content .....                 | 243 |
| 7.14.10 | DOM events for A/V Control object .....                                                         | 247 |
| 7.14.11 | Playback of memory audio .....                                                                  | 248 |
| 7.14.12 | Extensions to A/V Control object for media queuing.....                                         | 250 |
| 7.14.13 | Extensions to A/V Control object for volume control.....                                        | 251 |
| 7.14.14 | Extensions to A/V Control object for resource management .....                                  | 251 |
| 7.15    | Miscellaneous APIs.....                                                                         | 252 |
| 7.15.1  | The application/oipfMDTF embedded object .....                                                  | 252 |
| 7.15.2  | The application/oipfStatusView embedded object .....                                            | 254 |
| 7.15.3  | The application/oipfCapabilities embedded object.....                                           | 255 |
| 7.15.4  | The Navigator class .....                                                                       | 256 |
| 7.15.5  | Debug print API .....                                                                           | 256 |
| 7.16    | Shared Utility classes and features .....                                                       | 256 |
| 7.16.1  | Base collections .....                                                                          | 256 |
| 7.16.2  | The Programme class .....                                                                       | 257 |
| 7.16.3  | The ProgrammeCollection class .....                                                             | 262 |
| 7.16.4  | The DisclInfo class .....                                                                       | 262 |
| 7.16.5  | Extensions for playback of selected media components.....                                       | 262 |
| 7.16.6  | Additional support for protected content.....                                                   | 266 |
| 7.17    | DLNA RUI remote control function APIs .....                                                     | 267 |
| 7.17.1  | General .....                                                                                   | 267 |



|        |                                                                                                          |     |
|--------|----------------------------------------------------------------------------------------------------------|-----|
| 7.17.2 | The application/oipfRemoteControlFunction embedded object .....                                          | 267 |
| 8      | System integration aspects.....                                                                          | 272 |
| 8.1    | HTTP protocol.....                                                                                       | 272 |
| 8.1.1  | General .....                                                                                            | 272 |
| 8.1.2  | HTTP User-Agent header .....                                                                             | 272 |
| 8.1.3  | HTTP X-OITF-RCF-User-Agent header .....                                                                  | 273 |
| 8.2    | Mapping from APIs to protocols .....                                                                     | 273 |
| 8.2.1  | General .....                                                                                            | 273 |
| 8.2.2  | CoD download over HTTP .....                                                                             | 274 |
| 8.2.3  | CoD unicast streaming with SIP session management .....                                                  | 274 |
| 8.2.4  | Scheduled content multicast streaming with SIP session management .....                                  | 278 |
| 8.2.5  | Communication services with SIP session management .....                                                 | 284 |
| 8.2.6  | CoD unicast streaming over RTP and HTTP .....                                                            | 284 |
| 8.2.7  | Scheduled content multicast streaming .....                                                              | 288 |
| 8.3    | URI schemes and their usage .....                                                                        | 289 |
| 8.3.1  | General .....                                                                                            | 289 |
| 8.3.2  | Media fragments support .....                                                                            | 290 |
| 8.4    | Mapping from APIs to content formats .....                                                               | 291 |
| 8.4.1  | Character conversion.....                                                                                | 291 |
| 8.4.2  | AVComponent .....                                                                                        | 291 |
| 8.4.3  | Channel.....                                                                                             | 294 |
| 8.4.4  | Programme, ScheduledRecording, Recording and Download .....                                              | 299 |
| 8.4.5  | Exposing audio description streams as AVComponent objects .....                                          | 307 |
| 8.4.6  | HTML5 media element mapping.....                                                                         | 307 |
| 8.5    | DLNA RUI remote control function implementation .....                                                    | 309 |
| 8.5.1  | General .....                                                                                            | 309 |
| 8.5.2  | Relationship between DAE application and control UI .....                                                | 309 |
| 8.5.3  | XML UI listing provisioning .....                                                                        | 310 |
| 8.5.4  | Retrieving the control UI .....                                                                          | 312 |
| 8.5.5  | Receiving and responding to a message between the control UI in the remote control device and OITF ..... | 313 |
| 8.5.6  | Notification to the remote control device .....                                                          | 315 |
| 8.5.7  | Handling multiple DAE applications and multiple remote control devices .....                             | 316 |
| 9      | Capabilities .....                                                                                       | 317 |
| 9.1    | Minimum DAE capability requirements .....                                                                | 317 |
| 9.1.1  | General .....                                                                                            | 317 |
| 9.1.2  | SSL/TTLS Requirements .....                                                                              | 320 |
| 9.2    | Default UI profiles .....                                                                                | 321 |
| 9.3    | Client capability description .....                                                                      | 324 |
| 9.3.1  | General .....                                                                                            | 324 |
| 9.3.2  | Tuner/broadcast capability indication .....                                                              | 325 |
| 9.3.3  | Broadcast content over IP capability indication .....                                                    | 326 |
| 9.3.4  | PVR capability indication .....                                                                          | 326 |
| 9.3.5  | Download CoD capability indication .....                                                                 | 327 |
| 9.3.6  | Parental ratings .....                                                                                   | 328 |
| 9.3.7  | Extended A/V API support .....                                                                           | 329 |
| 9.3.8  | OITF metadata API support .....                                                                          | 329 |
| 9.3.9  | OITF configuration API support.....                                                                      | 329 |
| 9.3.10 | Communication services API Support .....                                                                 |     |

|                       |                                                                                           |     |
|-----------------------|-------------------------------------------------------------------------------------------|-----|
| 9.3.11                | DRM capability indication .....                                                           | 330 |
| 9.3.12                | Media profile capability indication .....                                                 | 331 |
| 9.3.13                | Remote diagnostics support.....                                                           | 332 |
| 9.3.14                | SVG .....                                                                                 | 332 |
| 9.3.15                | Third party notification support .....                                                    | 333 |
| 9.3.16                | Multicast delivery terminating function support.....                                      | 333 |
| 9.3.17                | Other capability extensions.....                                                          | 333 |
| 9.3.18                | HTML5 video .....                                                                         | 333 |
| 9.3.19                | DLNA RUI remote control function support .....                                            | 333 |
| 9.3.20                | Power consumption .....                                                                   | 333 |
| 9.3.21                | Widgets .....                                                                             | 333 |
| 9.3.22                | Buffer control of AV content playback API support.....                                    | 334 |
| 9.3.23                | Temporal clipping .....                                                                   | 334 |
| 9.3.24                | Capability elements from other schemas.....                                               | 335 |
| 9.3.25                | Pointer support.....                                                                      | 335 |
| 10                    | Security.....                                                                             | 335 |
| 10.1                  | Application / service security.....                                                       | 335 |
| 10.1.1                | General .....                                                                             | 335 |
| 10.1.2                | OITF requirements.....                                                                    | 335 |
| 10.1.3                | Server requirements .....                                                                 | 336 |
| 10.1.4                | Specific security requirements for privileged JavaScript APIs .....                       | 336 |
| 10.1.5                | Permission names .....                                                                    | 339 |
| 10.1.6                | Loading documents from different domains.....                                             | 341 |
| 10.2                  | User authentication.....                                                                  | 341 |
| 10.3                  | DLNA RUI remote control.....                                                              | 341 |
| 11                    | DAE Widgets .....                                                                         | 341 |
| 11.1                  | General.....                                                                              | 341 |
| 11.2                  | Widgets packaging and configuration .....                                                 | 341 |
| 11.3                  | Access request .....                                                                      | 342 |
| 11.4                  | Widget interface.....                                                                     | 342 |
| 11.5                  | Digital signature.....                                                                    | 342 |
| 12                    | Graphics performance .....                                                                | 343 |
| 12.1                  | Overview.....                                                                             | 343 |
| 12.2                  | Performance levels .....                                                                  | 343 |
| 12.3                  | Minimum 2D graphics performance .....                                                     | 343 |
| 12.4                  | Minimum 3D graphics performance .....                                                     | 344 |
| 12.5                  | Minimum canvas performance.....                                                           | 344 |
| 12.6                  | Minimum WebGL performance .....                                                           | 344 |
| 12.7                  | Performance measurement .....                                                             | 344 |
| Annex A (informative) | Design rationale – application model .....                                                | 346 |
| Annex B (informative) | Clarification of download CoD, streaming CoD and CSP interfaces.....                      | 347 |
| B.1                   | Overview.....                                                                             | 347 |
| B.2                   | List of interfaces .....                                                                  | 348 |
| B.2.1                 | Interface a): browse, select and purchase content from CoD store.....                     | 348 |
| B.2.2                 | Interface b*): in-session interaction from web page with underlying DRM agent .....       | 348 |
| B.2.3                 | Interface c*): autonomous out-of-session interaction between DRM agent and CoD store..... | 348 |

|                       |                                                                                                       |     |
|-----------------------|-------------------------------------------------------------------------------------------------------|-----|
| B.2.4                 | Interface d*): downloading content.....                                                               | 349 |
| B.2.5                 | Interface e*): unicast streaming and playback of downloaded content<br>using A/V Control object ..... | 351 |
| B.2.6                 | Interface f): request licence .....                                                                   | 351 |
| B.2.7                 | Interface g*): local metadata based applications .....                                                | 351 |
| B.3                   | Additional notes about content-on-demand .....                                                        | 351 |
| Annex C (normative)   | Content access descriptor syntax and semantics .....                                                  | 352 |
| C.1                   | Content access download descriptor format .....                                                       | 352 |
| C.2                   | Content access streaming descriptor format.....                                                       | 353 |
| C.3                   | Abstract content access descriptor format.....                                                        | 354 |
| Annex D (normative)   | Capability extensions schema .....                                                                    | 359 |
| Annex E (normative)   | Client channel listing format .....                                                                   | 362 |
| Annex F (normative)   | Display model.....                                                                                    | 366 |
| F.1                   | Logical plane model.....                                                                              | 366 |
| F.2                   | Interaction with the video/broadcast and A/V Control objects .....                                    | 367 |
| F.3                   | Graphic safe area .....                                                                               | 368 |
| F.4                   | Current channel .....                                                                                 | 368 |
| Annex G (normative)   | Backwards compatible profile of HTML5 media elements .....                                            | 370 |
| G.1                   | General.....                                                                                          | 370 |
| G.2                   | Video element.....                                                                                    | 370 |
| G.3                   | Audio element.....                                                                                    | 370 |
| G.4                   | Source element.....                                                                                   | 371 |
| G.5                   | Media element .....                                                                                   | 371 |
| G.6                   | Other object types.....                                                                               | 372 |
| G.7                   | Dependencies .....                                                                                    | 372 |
| Annex H (informative) | DLNA RUI remote control function sequences .....                                                      | 373 |
| H.1                   | Remote UI and box models .....                                                                        | 373 |
| H.1.1                 | Overview .....                                                                                        | 373 |
| H.1.2                 | i-box model.....                                                                                      | 374 |
| H.1.3                 | 2-box model.....                                                                                      | 374 |
| H.1.4                 | 3-box model.....                                                                                      | 375 |
| H.2                   | DLNA RUI remote control function sequences.....                                                       | 375 |
| H.2.1                 | General .....                                                                                         | 375 |
| H.2.2                 | Launching a DAE application to obtain the Control UI .....                                            | 376 |
| H.2.3                 | Obtaining the control UI from a running DAE application.....                                          | 378 |
| H.2.4                 | Sending and receiving messages between the remote control device<br>and DAE application .....         | 380 |
| Annex I (normative)   | Collections .....                                                                                     | 382 |
| I.1                   | General.....                                                                                          | 382 |
| I.2                   | The Collection template .....                                                                         | 382 |
| I.2.1                 | General .....                                                                                         | 382 |
| I.2.2                 | Properties.....                                                                                       | 382 |
| I.2.3                 | Methods .....                                                                                         | 382 |
| Annex J (informative) | SVG video tag support.....                                                                            | 383 |
| Annex K (informative) | Multimedia telephony sequences.....                                                                   | 386 |
| K.1                   | General.....                                                                                          | 386 |
| K.2                   | Full-duplex voice telephony call flow .....                                                           | 386 |
| K.3                   | Full-duplex video telephony call flow .....                                                           | 386 |

|                       |                                                                                                                                                    |     |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| K.4                   | Capture device and call parameters setting flow .....                                                                                              | 389 |
| K.5                   | Full-duplex Voice to Video telephony session update flow.....                                                                                      | 390 |
| Annex L (informative) | Server root certificate selection policy .....                                                                                                     | 392 |
| L.1                   | Overview.....                                                                                                                                      | 392 |
| L.2                   | Background.....                                                                                                                                    | 392 |
| L.3                   | Policy.....                                                                                                                                        | 392 |
| Annex M (normative)   | Changes to 5.6.2 of CEA-2014-A .....                                                                                                               | 394 |
| Bibliography          | .....                                                                                                                                              | 397 |
| Figure 1              | – OITF architecture .....                                                                                                                          | 20  |
| Figure 2              | – OIPF architecture with DLNA RUI RCF scenario .....                                                                                               | 37  |
| Figure 3              | – State diagram of OITF power states .....                                                                                                         | 41  |
| Figure 4              | – Behaviour when the selected channel changes .....                                                                                                | 47  |
| Figure 5              | – Behaviour when the application signalling for the currently selected channel changes or when a running broadcast-related application exits ..... | 49  |
| Figure 6              | – General event notification architecture on OITF and remote UI server .....                                                                       | 57  |
| Figure 7              | – HNI-IGI transaction for outgoing SIP requests from a DAE application .....                                                                       | 60  |
| Figure 8              | – HNI-IGI transaction for in-session incoming SIP request .....                                                                                    | 62  |
| Figure 9              | – What happens when the OITF is first turned on .....                                                                                              | 64  |
| Figure 10             | – User logs in using the DAE interface .....                                                                                                       | 65  |
| Figure 11             | – Unsolicited message from the network .....                                                                                                       | 66  |
| Figure 12             | – State diagram for embedded application/oipfDownloadManager objects .....                                                                         | 105 |
| Figure 13             | – State machine for a metadata search .....                                                                                                        | 176 |
| Figure 14             | – State diagram for embedded video/broadcast objects .....                                                                                         | 183 |
| Figure 15             | – PVR States for recordNow and timeshifting using video/broadcast .....                                                                            | 199 |
| Figure 16             | – State diagram for embedded A/V Control objects (normative).....                                                                                  | 236 |
| Figure 17             | – XML UI listing provisioning .....                                                                                                                | 310 |
| Figure 18             | – Remote control message handling.....                                                                                                             | 313 |
| Figure 19             | – Remote control device changes mapping between DAE applications .....                                                                             | 317 |
| Figure 20             | – Remote control device retains control of DAE application .....                                                                                   | 317 |
| Figure B.1            | – Main scenario .....                                                                                                                              | 347 |
| Figure F.1            | – Logical plane model .....                                                                                                                        | 366 |
| Figure F.2            | – Graphic safe area .....                                                                                                                          | 368 |
| Figure H.1            | – i-box model.....                                                                                                                                 | 374 |
| Figure H.2            | – 2-box Model.....                                                                                                                                 | 375 |
| Figure H.3            | – 3-box model.....                                                                                                                                 | 375 |
| Figure H.4            | – Launching of a DAE application .....                                                                                                             | 377 |
| Figure H.5            | – Obtaining remote control of a running DAE application .....                                                                                      | 379 |
| Figure H.6            | – Message flow between the remote control device and the DAE application .....                                                                     | 381 |
| Figure K.1            | – Full-duplex voice telephony call flow .....                                                                                                      | 387 |
| Figure K.2            | – Full-duplex Video telephony call flow .....                                                                                                      | 388 |
| Figure K.3            | – Capture device and call parameters setting flow .....                                                                                            | 390 |
| Figure K.4            | – Full-duplex Voice to Video telephony session update flow.....                                                                                    | 391 |

|                                                                                                         |     |
|---------------------------------------------------------------------------------------------------------|-----|
| Table 1 – Events applicable for cross application event handling .....                                  | 29  |
| Table 2 – Application signalling.....                                                                   | 51  |
| Table 3 – DAE application control codes .....                                                           | 53  |
| Table 4 – Supported application signalling features .....                                               | 53  |
| Table 5 – Key to status column .....                                                                    | 55  |
| Table 6 – New DOM events for application support .....                                                  | 83  |
| Table 7 – Metadata search states .....                                                                  | 177 |
| Table 8 – State transitions for the video/broadcast embedded object .....                               | 184 |
| Table 9 – Properties of the A/V Control object when the type attribute refers to video or audio .....   | 232 |
| Table 10 – Additional properties of the A/V Control object when the type attribute refers to video..... | 233 |
| Table 11 – Methods of the A/V Control object when the type attribute refers to video or audio .....     | 234 |
| Table 12 – Additional methods of the A/V Control object when the type attribute refers to video.....    | 234 |
| Table 13 – Additional applicable requirements from CEA-2014 .....                                       | 235 |
| Table 14 – URI schemes and usages .....                                                                 | 290 |
| Table 15 – Base UI profile names .....                                                                  | 321 |
| Table 16 – Complementary UI profile name fragments .....                                                | 322 |
| Table 17 – Minimum 2D graphics performance.....                                                         | 344 |
| Table F.1 – Clarification of the "current channel" concept in different scenarios .....                 | 369 |
| Table J.1 – SVG video tag support.....                                                                  | 383 |

## INTERNATIONAL ELECTROTECHNICAL COMMISSION

# CONSUMER TERMINAL FUNCTION FOR ACCESS TO IPTV AND OPEN INTERNET MULTIMEDIA SERVICES –

## Part 5-1: Declarative application environment

### FOREWORD

- 1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
- 3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
- 5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
- 8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
- 9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 62766-5-1 has been prepared by IEC technical committee 100: Audio, video and multimedia systems and equipment.

The text of this standard is based on the following documents:

| CDV          | Report on voting |
|--------------|------------------|
| 100/2548/CDV | 100/2662/RVC     |

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives, part 2.

A list of all parts in the IEC 62766 series, published under the general title *Consumer terminal function for access to IPTV and open internet multimedia services*, can be found on the IEC website.

In this standard, the following print type is used: object and event labels: Lucida Console.

The committee has decided that the contents of this publication will remain unchanged until the stability date indicated on the IEC web site under "<http://webstore.iec.ch>" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

A bilingual version of this publication may be issued at a later date.

**IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.**

## INTRODUCTION

The IEC 62766 series is based on a series of specifications that was originally developed by the OPEN IPTV FORUM (OIPF). They specify the user-to-network interface (UNI) for consumer terminals to access IPTV and open internet multimedia services over managed or non-managed networks as defined by OIPF.

# CONSUMER TERMINAL FUNCTION FOR ACCESS TO IPTV AND OPEN INTERNET MULTIMEDIA SERVICES –

## Part 5-1: Declarative application environment

### 1 Scope

This part of IEC 62766 specifies the Declarative Application Environment (DAE) component of the OIPF terminal function (OITF). The DAE is a declarative language based environment (browser) based on the OIPF web standards TV profile specified in IEC 62766-5-2 for the presentation of user interfaces and including scripting support for interaction with network server-side applications and access to the APIs of the other OITF functions.

### 2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 62481, *Digital living network alliance*

IEC 62766-1, *Open IPTV Forum (OIPF) consumer terminal function and network interfaces for access to IPTV and open Internet multimedia services – Part 1: General*

IEC 62766-2-1, *Open IPTV Forum (OIPF) consumer terminal function and network interfaces for access to IPTV and open Internet multimedia services – Part 2-1: Media Formats*

IEC 62766-2-2, *Open IPTV Forum (OIPF) consumer terminal function and network interfaces for access to IPTV and open Internet multimedia services – Part 2-2: HTTP Adaptive Streaming*

IEC 62766-3:2016, *Open IPTV Forum (OIPF) consumer terminal function and network interfaces for access to IPTV and open Internet multimedia services – Part 3: Content Metadata*

IEC 62766-4-1:2017, *Open IPTV Forum (OIPF) consumer terminal function and network interfaces for access to IPTV and open Internet multimedia services – Part 4-1: Protocols*

IEC 62766-5-2, *Open IPTV Forum (OIPF) consumer terminal function and network interfaces for access to IPTV and open Internet multimedia services – Part 5-2: Web standards TV profile*

IEC 62766-7:2017, *Open IPTV Forum (OIPF) consumer terminal function and network interfaces for access to IPTV and open Internet multimedia services – Part 7: Authentication, content protection and service protection*

ISO/IEC 15938-5:2003, *Multimedia content description interface – Part 5: Multimedia description schemes*

ISO/IEC 15948:2004, *Information technology – Computer graphics and image processing – Portable Network Graphics (PNG): Functional specification*

ISO/IEC 23009-1:2014, *Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats*

ISO 639-2, *Codes for the representation of names of languages – Part 2: Alpha-3 code*

ETSI TS 102 034 V1.5.1, *Digital Video Broadcasting (DVB); Transport of MPEG-2 TS Based DVB Services over IP Based Networks (and associated XML)*

ETSI TS 102 323, *Digital Video Broadcasting (DVB); Carriage and signalling of TV-Anytime information in DVB transport streams*

ETSI TS 102 539, *Digital Video Broadcasting (DVB); Carriage of Broadband Content Guide (BCG) information over Internet Protocol (IP)*

ETSI TS 102 809, *Digital Video Broadcasting (DVB); Signalling and carriage of interactive applications and services in Hybrid broadcast/broadband environments*

ETSI TS 102 851, *Digital Video Broadcasting (DVB); Uniform Resource Identifiers (URI) for DVB Systems*

ETSI TS 183 063, *Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); IMS-based IPTV stage 3 specification*

ETSI EN 300 468, *Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB Systems*

IETF RFC 1321, *The MD5 Message-Digest Algorithm*

IETF RFC 1918, *Address Allocation for Private Internets*

IETF RFC 2246, *The Transport Layer Security (TLS) Protocol Version 1.0*

IETF RFC 2326, *Real Time Streaming Protocol (RTSP)*

IETF RFC 2616, *Hypertext Transfer Protocol – HTTP/1.1*

IETF RFC 2818, *HTTP over TLS*

IETF RFC 3550, *RTP: A Transport Protocol for Real-Time Applications*

IETF RFC 3840, *Indicating User Agent Capabilities in the Session Initiation Protocol (SIP)*

IETF RFC 3841, *Caller Preferences for the Session Initiation Protocol (SIP)*

IETF RFC 5019, *The Lightweight Online Certificate Status Protocol (OCSP) Profile for High-Volume Environments*

IETF RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2*

IETF RFC 5280, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*

IETF RFC 5746, *Transport Layer Security (TLS) Renegotiation Indication Extension*

IETF RFC 6265, *HTTP State Management Mechanism*

IETF RFC 6454, *The Web Origin Concept, December 2011*

3GPP TS 24.229, *IP Multimedia Call Control Protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP) Stage 3 (Release 8)*

3GPP TS 26.234 V9.3.0 (2010-06), *Transparent end-to-end Packet-switched Streaming Service (PSS) Protocols and codecs (Release 9)*

Consumer Electronics Association CEA-2014-A (including the August 2008 Errata), *Web-based Protocol Framework for Remote User Interface on UPnP Networks and the Internet (Web4CE)*

*Graphics Interchange Format version 89a*

Available from <http://www.w3.org/Graphics/GIF/spec-gif89a.txt>

Open Mobile Alliance, *OMA-TS-Presence\_SIMPLE\_XDM-V1\_1-20080627-A, Presence XDM Specification*

Open Mobile Alliance, *OMA-TS-SIMPLE\_IM-V1\_0-20080820-D, Instant Messaging using SIMPLE*

W3C, *Web Storage, W3C Recommendation 30 July 2013*

Available at <http://www.w3.org/TR/2013/REC-webstorage-20130730/>

W3C, *Widget Access Request Policy, W3C Recommendation 7 February 2012*

Available at: <http://www.w3.org/TR/2012/REC-widgets-access-20120207/>

W3C, *Widget Interface, W3C Recommendation 31 October 2013*

Available at: <http://www.w3.org/TR/2013/REC-widgets-apis-20131031/>

W3C, *XML Digital Signatures for Widgets, W3C Recommendation 18 April 2013*

Available at: <http://www.w3.org/TR/2013/REC-widgets-digsig-20130418/>

W3C, *Packaged Web Apps (Widgets) – Packaging and XML Configuration (Second Edition), W3C Recommendation 27 November 2012*

Available at: <http://www.w3.org/TR/2012/REC-widgets-20121127/>

W3C, *Media Fragments URI 1.0 (basic), W3C Recommendation 25 September 2012*

Available at <http://www.w3.org/TR/2012/REC-media-frags-20120925/>

### 3 Terms, definitions and abbreviated terms

#### 3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 62766-1 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at <http://www.electropedia.org/>
- ISO Online browsing platform: available at <http://www.iso.org/obp>

### 3.1.1

#### **audio from memory**

audible notifications and audio clips intended to be played from memory

### 3.1.2

#### **broadcast related application**

interactive application associated with a television or radio channel, with part of a television channel (e.g. a particular program or show) or other television content

Note 1 to entry: Often referred to as “red button” applications in the industry, regardless of how they are actually started by the end user.

### 3.1.3

#### **broadcast independent application**

interactive application not related to any TV channel or TV content or to the currently selected service provider

### 3.1.4

#### **control UI**

remote UI that controls DAE applications in the OITF, sent from an IPTV applications server via the OITF or pre-stored in the OITF, and rendered in the DLNA RUIC on the remote control device

### 3.1.5

#### **DLNA RUIC**

DLNA device with the role of finding and loading remote UI content exposed by a DLNA RUIS capability and rendering and interacting with the UI content

Note 1 to entry: This terminology references the DLNA RUI specification (IEC 62481-6-1).

### 3.1.6

#### **DLNA RUIS**

DLNA function in the OITF with the role of exposing and sourcing UI content

Note 1 to entry: This terminology references the DLNA RUI specification (IEC 62481-6-1).

### 3.1.7

#### **embedded object**

software module that extends the capabilities of the OITF browser

Note 1 to entry: Features provided by an embedded object are made available to DAE applications through the methods and properties of a specific JavaScript<sup>1</sup> object.

### 3.1.8

#### **HTML document**

XHTML document and associated style and script files conforming to the restrictions and extensions defined in this document

### 3.1.9

#### **key event**

event sent to a DAE application in response to input from the end-user

<sup>1</sup> JavaScript is a programming language that has been standardised by ECMA as ECMAScript® and as ISO/IEC 16262. This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.

Note 1 to entry: This input is typically generated in response to the end-user pressing a button on a conventional remote control. It may also be generated by some other mechanism on alternative input devices such as game controllers, touch-screens, wands or drastically reduced remote controls.

### 3.1.10

#### **non-visual embedded object**

embedded object that has no visible representation and cannot get input focus

### 3.1.11

#### **remote control device**

mobile or portable device which has the functionality of the DLNA RUIC

### 3.1.12

#### **remote UI**

display of a UI from one device on a second (remote) device across a network

### 3.1.13

#### **service provider related application**

interactive application related to the service provider selected through the service provider selection process

### 3.1.14

#### **trick mode**

facility to allow the user to control the playback of content, such as pause, fast and slow playback, reverse playback, instant access, replay, forward and reverse skipping

## 3.2 Abbreviated terms

For the purposes of this document, the abbreviations given in IEC 62766-1, as well as the following apply.

|        |                                  |
|--------|----------------------------------|
| AJAX   | Asynchronous JavaScript and XML  |
| CRID   | Content Reference Identifier     |
| DOM    | Document Object Model            |
| GIF    | Graphics Interchange Format      |
| HAS    | HTTP Adaptive Streaming          |
| HE-AAC | High Efficiency AAC              |
| IR     | Infra-Red                        |
| JPEG   | Joint Photographic Experts Group |
| MPD    | Media Presentation Description   |
| PNG    | Portable Network Graphics        |
| PSI    | Public Service Identifier        |
| RCF    | Remote Control Function          |
| SVG    | Scalable Vector Graphics         |
| TLS    | Transport Layer Security         |
| WAVE   | Waveform audio format            |

## 4 DAE overview

### 4.1 General

This document builds on a selection of W3C specifications defined in IEC 62766-5-2 with additions defined in this part of IEC 62766-5 in order to expose to an IPTV service provider the capabilities of the OITF. In clauses and subclauses of this document whose presence is

indicated by one of the capabilities defined in 9.3, the use of the terms "shall" or "required" applies only when the capability is made available to DAE applications. They do not have the effect of making that subclause mandatory. In clauses and subclauses of this document that are not covered by capabilities, terms such as "shall" apply as used in each clause and subclause.

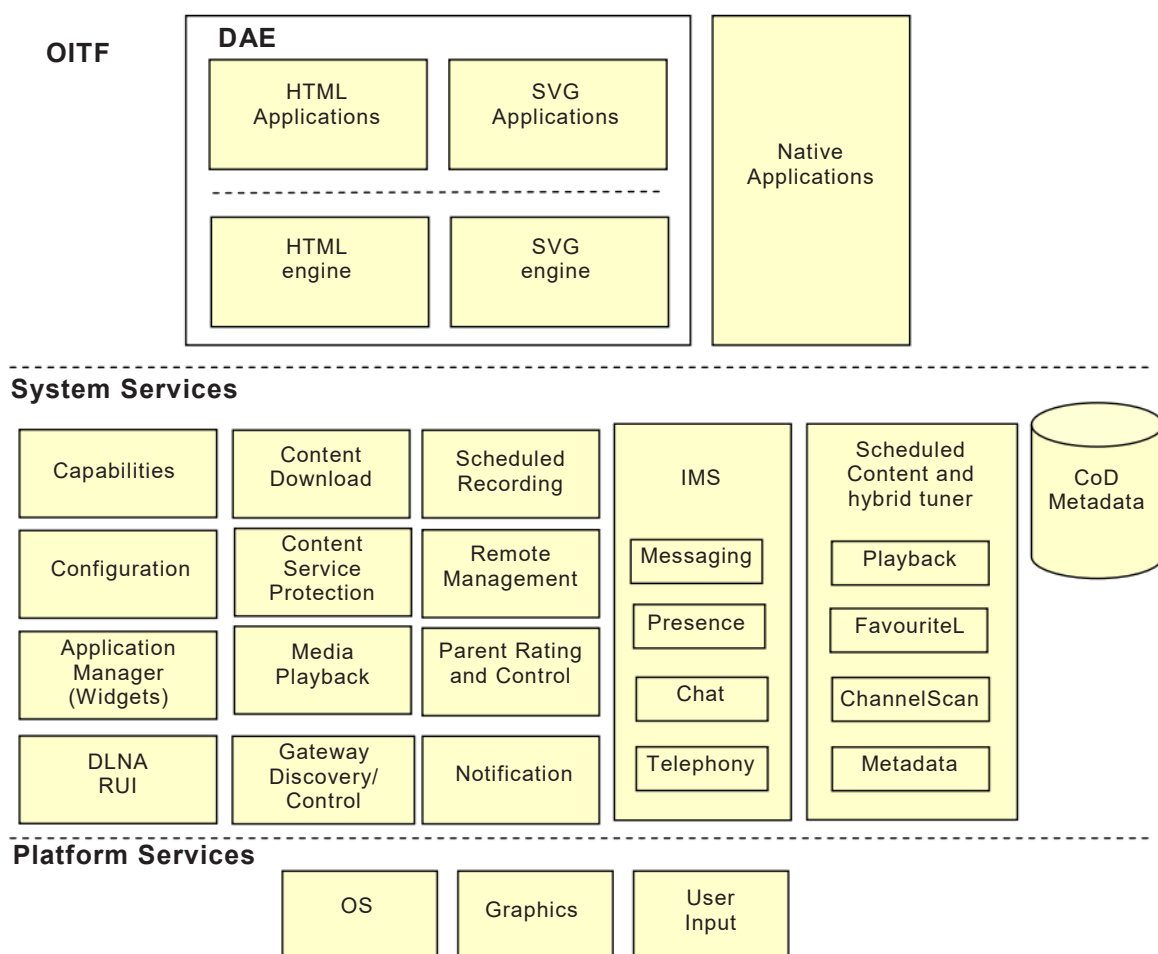
In this document, "application" means "declarative application" (browser-based application) throughout the DAE platform specification, as opposed to the "procedural applications" (Java-based applications), defined in IEC 62766-6, the PAE platform specification.

In the documented APIs, JavaScript attributes are read-write unless otherwise specified.

The type "Integer" is not a valid JavaScript type as is. It is used as a short hand notation for a subset of type "Number", which includes only the numbers that can be written without a fractional or decimal component.

## 4.2 Architecture of the DAE

Figure 1 provides an overview of the OITF architecture in relation to this document.



IEC

**Figure 1 – OITF architecture**

The various system services are described below:

**Application Manager (Widgets):** this service handles the starting and stopping of applications and the downloading, starting, stopping and removal of widgets. See 4.4, 5.1, 5.2.8, 7.2 and 11 .

**Capabilities:** this service handles terminal capabilities and exposing them to applications. See 7.15.3 and 9.3 .

**CoD metadata:** this service handles the downloading, storage and retrieval of CoD metadata. See 7.5 .

**Configurations:** this service handles reporting and changing device configuration and power management. See 7.3 .

**Content Download:** this service handles initiating downloading of content by applications, downloading the content and managing content once downloaded. See 4.7.2 and 7.4.

**Content Service Protection:** this service handles content and service protection. See 7.6.

**DLNA RUI:** this service enables a DAE application on an OITF to export a user interface to another device in the home as defined by the DLNA remote UI specification. See 4.10, 7.17 and 8.5.

**Gateway Discovery/Control:** this service handles gateways, including discovery and managing information about them. See 4.3 and 7.7.

**IMS – Messaging, Presence, Chat, Telephony:** this service handles IMS including messaging, presence, chat and telephony. See 7.8.

**Media Playback:** this service handles playback of media including streaming on-demand, downloaded content, and scheduled content that has been recorded. Live scheduled content is covered in a different service. See 4.8.2 and 7.14.

**Notification:** this service handles notifications from the network to the OITF. See 5.3.

**Parental Rating and Control:** this service handles parental rating and control including reporting of changing of parental rating status and providing parental rating PIN codes. See 4.6, 7.9, 7.13.6 and 7.14.6.

**Remote Management:** this service handles remote management when supported as a DAE application. See 7.11.

**Scheduled Content and hybrid tuner – playback, favourite lists, channel scan, metadata:** this service handles scheduled content services whether these are delivered by IP or by a classical cable, satellite or terrestrial tuner in a hybrid device. See 4.8.2.3, 4.9, 7.12 and 7.13.

**Scheduled Recording:** this service handles recording of scheduled content. See 7.10.

NOTE Native applications are out of scope of the DAE specification.

### **4.3 Gateway discovery and control**

This subclause describes how DAE applications discover the information of the gateway and subsequently interacts with the gateway. The discovery of the IG and AG by the OITF are defined in 10.1 of IEC 62766-4-1:2017. The discovery takes place prior to the DAE application being initialized. The information about the discovered gateways is made available to DAE applications through the `application/oipfGatewayInfo` embedded object. DAE applications can use this gateway information to interact with the discovered gateways (e.g.

IG, AG, CSP Gateway and so on). The `application/oipfGatewayInfo` embedded object shall be made accessible through the DOM with the interface as defined in 7.7.2.

Access to the functionality of the `application/oipfGatewayInfo` embedded object is privileged and shall adhere to the security requirements defined in 10.1.

## **4.4 Application definition**

### **4.4.1 General**

This subclause defines what is meant by the concept of a DAE application, which files and assets are considered to be part of a DAE application, and how this relates to DAE application security and lifecycle.

A DAE application is either:

- an associated collection of documents (HTML or SVG where supported) from within a common boundary (see 5.1.4 for more details), or
- a Widget as specified in 4.4.10.

While the application is loaded within the browser, an additional browser object (the `oipfApplicationManager` object) defined in 7.2.2 is present and accessible by the DAE application. The `oipfApplicationManager` object provides access to the `Application` class defined in 7.2.3.

The difference between a DAE application and a traditional web page is that web pages are stand-alone with no formal concept of a group of pages or a context within which a group of pages are loaded and execute. For this reason, the definition and details of a DAE application focuses on the application execution environment and the additional capabilities provided to DAE applications. Subclauses 4.4.2 to 4.4.11 describe some of the differences. Additional details about the DAE application lifecycle can be found in 5.1.

### **4.4.2 Similarities between applications and traditional web pages**

DAE applications are comprised of pages that are conceptually no different from traditional web pages. Both pages in a DAE application and traditional web pages can include the contents of other documents. These included documents can have a variety of types, including Cascading Style Sheets (CSS), JavaScript, SVG, JPEG, PNG and GIF.

A dynamic DOM, combined with XMLHttpRequest, permits AJAX-style changes to the current page in a DAE application or web page without necessarily replacing the entire document.

### **4.4.3 Differences between applications and traditional web pages**

A DAE application provides shared context and state common to a number of pages – a concept which doesn't formally exist in the web. Loading and unloading pages within the context of a DAE application is the same as loading and unloading web pages.

The application context includes information about the state of an application from the platform's perspective – permissions, priority (for example, which to terminate first in the event of insufficient resources) and similar information that spans all documents within an application during the lifetime of that application.

An OITF may support the execution of more than one application simultaneously. Applications may share the same screen estate in a defined and controlled fashion. This differs from multiple web pages, which are typically handled through different browser “windows” or “tabs” and may not share the same screen estate concurrently (although the details of this behaviour are often browser-dependent). This also differs from the use of frames, which, apart from iframes, do not support overlapping screen estate. Where simultaneous execution is required

than one application is supported, both foreground and background applications shall be supported simultaneously.

Where simultaneous execution of more than one application is supported, applications shall be recorded within a hierarchy of applications. Each object representing an application possesses an interface that provides access to methods and attributes that are uniquely available to applications. For example, facilities to create and destroy applications can be accessed through such methods.

#### 4.4.4 The application tree

Where simultaneous execution of more than one application is supported, applications are organised into a tree structure. Using the `createApplication()` method as defined in 7.2.3.3, applications can either be started as child nodes of the application or as a sibling of the application (i.e. added as an additional child of this application's parent). The root node of an application tree is created upon loading an initial application URI or by creating a sibling of an application tree's root node. An OITF may keep track of multiple application trees. Each of these individual application trees are connected to a hidden system root node maintained by the OITF that is not accessible by other applications.

Applications created while the DAE environment is running (e.g. as a result of an external notification) that are not created through `createApplication()` shall be created as children of the hidden system root node.

#### 4.4.5 The application display model

##### 4.4.5.1 General

Applications shall be displayed on the OITF in one of the application visualization modes as defined in 4.5.7.

The mode used shall be determined prior to initialisation of the DAE execution environment and shall persist until termination or re-initialization of the DAE execution environment. The means by which this mode is chosen is outside the scope of this document.

Each application has an associated DOM window object and a DOM document object that represents the document that is currently loaded for that application. Even “windowless” applications that are never made visible have an associated DOM Window object.

##### 4.4.5.2 Manipulating an application's DOM Window object

Standard DOM window methods are used to resize, scroll, position and access the application document (see 4.5.7). Many browsers restrict the size or location of windows; these restrictions shall NOT be enforced for windows associated with applications within the browser area. Any area of the display available to DAE applications may be used by any application. Thus, ‘widget’-style applications can create a small window that contains only the application without needing to be concerned with any minimum size restrictions enforced by browsers.

#### 4.4.6 The security model

Each application has a set of permissions to perform various privileged operations within the OITF. The permissions that are granted to an application are defined by the intersection of three permission sets:

- 1) the permissions requested by the application, using the mechanism defined in Clause 10;
- 2) the permissions supported by the OITF; some permissions may not be supported due to capability restrictions (e.g. the *permission\_pvr* permission will never be granted on a receiver that does not support PVR capability);

- 3) the permissions that may be granted, as determined by user settings or configuration settings specified by the operator (e.g. blacklists or whitelists; see Clause 10 for more information); this is a subset of 2), and may be different for different users.

#### **4.4.7 Inheritance of permissions**

Applications created by other applications (e.g. using the methods described in 5.1.2.2 or 5.1.2.4) shall not inherit the permissions issued to the parent application. The permissions granted to the new application will be defined by the mechanism specified in Clause 10.

When an application uses cross-document messaging (see the `window.postMessage()` method defined in HTML5 Web Messaging as referenced in IEC 62766-5-2 to communicate with another application, any action carried out in response to the message shall take place in the security context of the application to which the message was sent. Applications should take care to ensure that privileged actions are only taken in response to messages from an appropriate source.

#### **4.4.8 Privileged application APIs**

##### **4.4.8.1 General**

The privilege model implemented with applications is based upon requiring access to the `Application` object representing an application in order to access the privileged functionality related to application lifecycle management and inter-application communication.

##### **4.4.8.2 Compromising the security**

Since applications have access to `Application` objects, it is possible for applications to compromise the security of the framework by passing these objects to untrusted code. For example, an application could raise an event on an untrusted document and pass a reference to its `Application` object in the message. Where simultaneous execution of more than one application is supported, any calls to methods on an `Application` object from pages not running as part of an application from the same provider shall throw an error as defined in 10.1.2.

##### **4.4.9 Active applications list**

Where simultaneous execution of more than one application is supported, the OITF shall maintain a list of application nodes ordered in a "most recently activated" order – the active applications list. This list is used by the cross-application event dispatch algorithm as defined in 4.5.8 and is not directly visible to applications.

An application is activated through calling the `activateInput()` method of the application node. This marks an application as active and shall insert the application at the start of the active application list (removing it from the list first if it is already present).

An application is deactivated through the `deactivateInput()` method of the application node. This marks an application as inactive and shall remove it from the active application list.

The currently active application is the application at the start of the active application list.

This document does not define any behaviour if more than one copy of the browser is executing.

##### **4.4.10 Widgets**

DAE Widgets are a specialization of DAE applications and share aspects with W3C Widgets.

W3C Widgets are standardized by the “Widgets 1.0 family of specifications” as described in subclause 1.4 of W3C Widget Packaging and XML Configuration. Clause 11 specifies which parts of W3C Widgets specifications are supported by DAE Widgets. From here on, when using the word “Widget”, this means DAE Widgets as defined in this document.

Widgets can be primarily seen as packaged DAE applications. Since they are packaged, it is possible to have a single download and installation on an OITF. Widgets may also be installed on an OITF via non-HTTP distribution channels and even over off-network channels (e.g. a USB thumb drive). Packaging also provides an easy way to deploy and/or update applications on the OITF when it is installed in the home. The packaging and configuration of a DAE Widget is described in 11.2.

Since DAE Widgets are DAE Applications, everything that is defined for a DAE Application is also applicable to a Widget unless specified. Furthermore Widgets have several specific features as defined in Clause 11.

#### **4.4.11 Origin for broadcast-delivered documents**

For documents that are delivered by an object carousel in a broadcast channel (as defined in 4.1 of IEC 62766-2-1:2016, the following shall apply:

- for each broadcast channel, the OITF shall generate a unique origin as defined in IETF RFC 6454;
- these origins shall be of the form scheme/host/port tuple where the OITF does not use the same scheme for any documents other than those delivered by a broadcast object carousel;
- the origin shall be the one for the channel from which the document was loaded regardless of any subsequent channel changes.

Specifically, this origin shall be used with the Web Storage API as referenced in IEC 62766-5-2.

### **4.5 Resource management**

#### **4.5.1 General**

Subclause 4.5 describes how resources (including non-granular resources such as memory and display area) are shared between multiple applications that may be running simultaneously. Applications should be able to tolerate the loss of scarce resources if they are needed by another application, and should follow current industry best practices in order to minimize the resources they consume.

This document is silent about the mechanism for sharing resources between DAE applications and other applications running on the OITF. In the remainder of this subclause and this document, the term application refers solely to DAE applications.

#### **4.5.2 Application lifecycle issues**

Where simultaneous execution of more than one application is supported, if an application attempts to start and not enough resources are available, the application with the lowest priority may be terminated until sufficient resources are available for the new application to execute or until no applications with a lower priority are running. Applications without a priority associated with them (e.g. applications started by the DRM agent, see 5.1.2.7) shall be assumed to have a priority of 0x7F.

Applications may register a listener for ApplicationUnloaded events (see 7.2.2.5) to receive notification of the termination of a child application, where simultaneous execution of more than one application is supported.

Failure to load an asset (e.g. an image file) or CSS file due to a lack of memory shall have no effect on the lifecycle of an application, but may result in visual artefacts (e.g. images not being displayed). Failure to load an HTML file due to a lack of memory may cause the application to be terminated.

#### **4.5.3 Caching of application files**

Application files may be cached on the receiver in order to improve performance; this document is silent about the use of any particular caching strategy.

#### **4.5.4 Memory usage**

Applications should use current industry best practices to avoid memory leaks and to free memory when it is no longer required. In particular, applications shall unregister all event listeners before termination, and should unregister them as soon as they are no longer required.

Where available, applications shall use explicit destructor functions to indicate to the platform that resources may be re-used by other applications.

Applications may use the `gc()` method on the `application/oipfApplicationManager` embedded object to provide hints to the OITF that a garbage collection cycle should be carried out. The OITF is not required to act on these hints.

The `LowMemory` event described in 7.2.2.5 shall be generated when the receiver is running low on memory. The amount of free memory that causes this event to be generated is implementation dependent. Applications may register a listener for these events in order to handle low-memory situations as they choose best.

#### **4.5.5 Instantiating embedded objects and claiming scarce system resources**

The objects defined in Clause 7 are embedded objects. These are typically instantiated through the standard DOM 2 methods for creating HTML objects or the `oipfObjectFactory` as defined in 7.1.

All embedded objects as defined in Clause 7 shall not claim scarce system resources (such as a hybrid tuner) at the time of instantiation. Hence, instantiation shall not fail if the object type is supported (and sufficient memory is available).

For each embedded object for which scarce resource conflicts may be a problem, the state diagram and the accompanying text define how to deal with claiming (and releasing) scarce system resources.

Once an OIPF embedded object has been instantiated, dynamic change of its MIME type, which could cause the properties and methods associated with the object to change, shall be ignored.

For instance, it is possible to change the MIME type of an A/V Control embedded object from `video/mpeg` to `video/mp4` but it is not possible to change the MIME type of an OIPF embedded object from `"application/oipfApplicationManager"` to `"application/oipfConfiguration"`.

#### **4.5.6 Media control**

If insufficient resources are available to present the media, the attempt to play the media shall fail. For the video/broadcast object, this shall be indicated by a `ChannelChangeError` event with a value of 11 for the error state. For an A/V Control object, the error property shall take the value 3.

When the video/broadcast or A/V Control object either is instantiated in the DYNAMIC\_ALLOCATION model or transitions to the DYNAMIC\_ALLOCATION model, scarce resources such as a media decoder shall only be claimed following a call to the `bindToCurrentChannel()`, `setChannel()`, `nextChannel()` or `prevChannel()` methods on a video/broadcast object or the `play()` method on an A/V Control object. By implication, instantiating a video/broadcast or A/V Control object does not cause the media referred to by the object's data attribute to start playing immediately. See 7.13.2.2 for details of when scarce resources are released by a video/broadcast object and 7.14.2.2 when scarce resources are released by an A/V Control object.

Scarce resources can be claimed by the video/broadcast or A/V Control object at instantiation time by specifying the `requiredCapabilities` parameter. In this case, the STATIC\_ALLOCATION method is used and the scarce resources are held by the object until it is either destroyed or the `release()` method is called.

This part is intentionally silent about handling of resource use by embedded applications including scheduled recordings.

#### 4.5.7 Use of the display

A compliant OITF shall support at least one of the following application visualization modes for managing the display of applications.

- a) Multiple applications may be visible simultaneously, with each application having a full-screen window, with the OITF managing the focus. Setting parts of an application to be transparent shall cause the following to be visible except where the application has drawn UI elements:
  - firstly any applications with a lower Z-index,
  - secondly video (if the hardware supports overlay as per the `<overlay*>` elements defined in 9.2 for the capability profiles).

In this mode, applications from the same service provider that are intended to run simultaneously should take care to coordinate their use of the display in order to ensure that important UI elements are not obscured.

- b) Multiple applications may be visible simultaneously, with the OITF managing the size, position, visibility and focus between applications.
- c) Only one application is visible at any time; switching to a different application either hides the currently visible application (where simultaneous execution of more than one application is supported) or terminates the currently visible application (where simultaneous execution of more than one application is not supported). The mechanism for switching between applications is implementation-dependent. In this case, the `show()`, `hide()`, `activateInput()` and `deactivateInput()` methods of the Application object provide hints to the execution environment about whether the user should be notified that an application requires attention. The mechanism for notifying the user is outside the scope of this document.

Applications shall be created with an associated DOM window object, that covers the display area made available by the OITF to a DAE application. The size of the DOM window can be retrieved through properties `innerwidth` and `innerHeight` of the DOM window object.

Any areas of the browser area outside the DOM window that become visible when it is resized shall be transparent – any video (if the hardware supports overlay as per the `<overlay*>` elements defined in 9.2 for the capability profiles) or applications (if multiple applications can be visible simultaneously) with a lower Z-index will be visible except where the application has drawn UI elements.

Broadcast-related and service provider related applications shall initially be created as invisible to avoid screen flicker during application start-up. Once loaded (as shall be indicated through an `onload` event handler), the application then typically calls the `show()` method of

its parent `Application` object. Broadcast-independent applications shall initially be created as visible and need not call these methods.

If the application does not ever need to be visible, then its DOM window object will never be shown. In that case, the application should take steps to avoid being formatted to reduce computation and memory overheads. This is typically accomplished by setting the default CSS style of the document's `BODY` element to `visibility:hidden`.

Because all applications have associated DOM window objects, it is possible to make any application visible even if it is not normally intended to be visible. This is of particular benefit during debugging of hidden service type applications.

Application developers should explicitly set the background colour of the application `<body>` and `<html>` elements.

Setting the background colour to 'transparent' (e.g. using the CSS construct `html, body { background-color: Transparent; }`) will allow the underlying video to be shown for those areas of the screen that are not obscured by overlapping non-transparent (i.e. opaque) children of the `<body>` element.

Changing the visibility of an application by calling method `show()` or `hide()` on the `Application` object shall not affect its use of resources. The application still keeps running and listens to events unless the application gets deactivated (see 4.4.9) or destroyed (see 5.1.3).

#### **4.5.8 Cross-application event handling**

##### **4.5.8.1 General**

As defined in the DOM Level 3 Events specification as referenced in IEC 62766-5-2, standard DOM events are raised on a specific node within a single document. This document extends the event capability of the OITF through cross-application events handling, but does not change the DOM2 event model for dispatching events within documents. Where simultaneous execution of more than one application is supported, an OITF shall implement the cross-application events and cross-application event handling model described in this subclause.

An OITF shall implement the following cross-application event handling model. Cancelling the propagation of an event in any phase shall abort further raising of the event in subsequent phases: if an event is eligible for cross-application event handling (see below for more information) and is targeted at a node in the most recently activated application, then dispatch the event to that node using the standard DOM bubbling/capturing of events. Default actions normally taken by the browser upon receipt of an event shall be carried out at the end of this step, unless overridden using the existing DOM methods (i.e. using method `preventDefault()`).

If the cross-application event is not prevented from being propagated beyond the document root node of the application by using the existing DOM methods, the event is dispatched to other active applications in the application hierarchy using the active applications list described in 4.4.9. The OITF shall iterate over the applications in the active application list, from most recently activated to least recently activated, dispatching the event to the `Application` object of each application in turn. Note that the event shall NOT be dispatched to the document, and default browser action shall NOT be carried out during this phase. Cancelling the propagation of an event in this phase shall abort further raising of the event in subsequent applications.

Event listeners for cross-application events are registered and unregistered using the same mechanism as for DOM2 events. Listeners for cross-application events may be registered on the `Application` object as well as on nodes in the DOM tree.

The following events are valid instances of cross-application events and are applicable for cross application event handling, as shown in Table 1.

**Table 1 – Events applicable for cross application event handling**

| System event | Description                                                                                                           |
|--------------|-----------------------------------------------------------------------------------------------------------------------|
| KeyPress     | Generated when a key has been pressed by the user. May also be generated when a key is held down during a key-repeat. |
| KeyUp        | Generated when a key pressed by the user has been released.                                                           |
| KeyDown      | Generated when a key has been pressed by the user.                                                                    |

The KeyPress, KeyUp and KeyDown events are all targeted cross-application events. The events are targeted at the node that has the input focus.

All events dispatched using the standard `dispatchEvent()` method are normal DOM events, not cross-application events. As defined in HTML5 Web Messaging as referenced in IEC 62766-5-2, the OITF shall support the `window.postMessage()` method for cross-document messaging. The method takes two arguments; a message (of type String) to be dispatched and the `targetOrigin`, which defines the expected origin (i.e. domain) of the target window, or "\*" if the message can be sent to the target regardless of its origin. The target of the event is the "window" of a specific application. Applications can use this method to send events to other applications. The receiving application may receive those events and interpret them, or may dispatch them in its DOM using standard DOM `dispatchEvent()` methods.

The visibility of an application shall not affect the cross-application event handling algorithm as defined above – an active application shall receive cross-application events even when it is not visible.

Incoming key events are dispatched using the cross-application event handling algorithm as defined above.

**NOTE** This event dispatch model enables key events to be dispatched to multiple applications. Applications wishing to become the primary receiver for key events should call `Application.activateInput()`. Even though `Application.activateInput()` is called, another application may subsequently be activated. In order to ensure that sensitive key input (e.g. PINs or credit card details) is limited only to the application it is intended for, applications should check that they are the primary receiver of the key events (using the `Application.isPrimaryReceiver` property and/or the `ApplicationPrimaryReceiver` and `ApplicationNotPrimaryReceiver` events defined in 7.2.7) and should 'absorb' key events by calling the `stopPropagation()` method on the DOM2 key event.

#### 4.5.8.2 Behaviour of the BACK key

OIPF applications may use the methods on the History object to navigate the history list. The history list shall not go back beyond the initial page of an OIPF application.

If a remote features a "back" or "back up" key, or one offering similar functionality, the OITF shall handle this key as described below:

A `VK_BACK` key event shall be dispatched to applications following the normal key handling process described in 4.5.8.

The default behaviour of the `VK_BACK` key event is implementation-dependent but the OITF shall not load the previous page in its history list for DAE applications.

#### 4.5.9 Tuner resources

Tuners can be used for recording, scanning or watching broadcast channels (e.g. DVB-T). The priority relating to resource management is as follows. Recording has the highest priority.

viewing a channel has the lowest priority. A record request shall not be automatically interrupted by viewing a channel or a scan request. Note that to free the tuner for viewing requires interrupting the recording first.

#### 4.6 Parental access control

A number of different approaches to parental access control is permitted.

##### a) Enforcement in the network.

An IPTV service provider may manage parental access control completely in the network. Applications running on application servers back in the network may decide to block access to content or arrange a DAE application to ask for a PIN code as necessary. This approach can apply to any kind of content – streaming on-demand content, IP broadcast content and to downloaded content.

No specific support is needed for this approach in the specification.

##### b) Enforcement in the OITF CSP / CSPG for protected MPEG-2 TS content

IPTV service providers may use the content protection mechanism for protected content to enforce access control to protected content. If used, this enforcement will happen in the OITF and, in some cases, in the CSP Gateway as well. In this approach, the content protection mechanism in the OITF would ask for PIN codes as needed.

The OITF CSP/CSPG-based enforcement of this approach and link to DAE API and events are defined in:

- 4.2.6.2 of IEC 62766-7:2017, for CSP terminal centric approach,
- 4.3.3 and 4.3.4.5.1.1.3 of IEC 62766-7:2017 for CI+ CSP Gateway centric approach
- 4.3.3 and 4.3.5.6.1 of IEC 62766-7 for DTCP-IP CSP Gateway centric approach

##### c) Enforcement in the OITF

An OITF may enforce parental access controls itself. Examples include embedded applications offering access to:

- IP delivered content based on information delivered to the metadata CG client,
- classical broadcast content in hybrid OITFs,
- content delivered to the OITF (either streaming or downloaded).

In approaches b) and c), PIN dialogs would be generated by code forming part of the OITF implementation. The APIs in 7.9 provide some control over these dialogs. The PIN would typically be configured by an embedded application but may also be configured by a DAE application using the optional APIs defined in 7.3.3.

These approaches b) and c) are reflected in a number of failure modes as defined in the following subclauses of this document:

- for broadcast channels (both IP and hybrid), in 7.13.2.3, see `onChannelChangeError` where `errorState 3` is defined as "parental lock on channel";
- parental rating errors and parental rating changes during playback of A/V content through the AV Control APIs defined in 7.14 and the `video/broadcast` object are reported according to the mechanism described in 7.14.6 and 7.13.6 respectively.

NOTE Due to the variation in regulatory requirements and deployment scenarios, this standard is intentionally silent about which of these approaches or combination of approaches is used.

## 4.7 Content download

### 4.7.1 General

The requirements in this subclause apply if the `<download>` element has been given value "true" in the OITF's capability profile as specified in 9.3.5.

### 4.7.2 Download manager

An OITF shall support a native download manager (i.e. "Content Download" component) to perform the actual download and storage of the content, and which allows the user to manage (e.g. suspend/resume, cancel) and monitor the download, in a consistent manner across different service providers. The download manager shall continue downloading as a background process even if the browser does not have an active session with the server that originated the download request anymore (e.g. has switched to another DAE application), even after a device power-down or network failure, until it succeeds or the user has given permission to terminate the download (see 4.7.5 on HTTP Range support to resume HTTP downloads after a power/network failure).

The native download manager shall be able to offer a visualization of its status through the `application/oipfStatusView` embedded object as defined in 7.15.2.1.

If the attribute `manageDownloads` of the `<download>` element in the client capability description is unequal to "none", the native download manager shall offer control over the active downloads through the JavaScript API defined by the `application/oipfDownloadManager` embedded object in 7.4.4.

NOTE 1 Once sufficient data of the content has been downloaded, the content may be played back using a native application, and may be played back using an A/V Control object. In the latter case, see method `setSource()` in 7.14.8 for more information.

NOTE 2 Annex D clarifies the content download usage scenario in more detail.

### 4.7.3 Content access download descriptor

An OITF shall support parsing and interpretation of the content access download descriptor document format with the specified semantics, syntax and MIME type as specified in Annex E.

### 4.7.4 Triggering a download

#### 4.7.4.1 General

An OITF shall support a non-visual embedded object of type "application/oipfDownloadTrigger", with the JavaScript API as defined in 7.4.2 and 7.4.3 to trigger a download.

Subclauses 4.7.4.2 to 4.7.4.5 define some details about the different ways of triggering a download.

#### 4.7.4.2 Using the `registerDownload()` method

The `registerDownload()` method takes a content access download descriptor as one of its arguments and passes it to the underlying native download manager in order to trigger a download. The following requirements apply.

The content access download descriptor may be created in JavaScript or may be fetched using `XMLHttpRequest`. To this end, the OITF shall pass the data inside the content access download descriptor into the `XMLHttpRequest.responseXML` property in JavaScript for further processing, if the OITF encounters an HTTP response message with the Content-Type of "application/vnd.oipf.ContentAccessDownload+xml", as the result of an `XMLHttpRequest`.

NOTE The behaviour in other cases when the OITF encounters an HTTP response message with the Content-Type "application/vnd.oipf.ContentAccessDownload+xml", for example whilst following a link as specified by an anchor element (<a>), is not specified in this document.

If the OITF supports a DRM agent with a matching DRMSystemID as per 9.3.11, the OITF shall pass included DRM-information as part of the <DRMControlInformation> elements of a content-access download descriptor to the DRM agent.

If the content access descriptor contains multiple content items to be downloaded, then all items are considered to belong together. Therefore, the download of each individual content item has the same download identifier in that case (whereby the ContentID may be used for differentiation). The order by which the items are downloaded is defined by the OITF.

#### 4.7.4.3 Using the registerDownloadURL() method

The registerDownloadURL() method takes a URL as one of the arguments and passes it to the underlying native download manager in order to trigger a download. The URL may point to any type of content. The URL may also point to a content access download descriptor (i.e. with argument contentType having value "application/vnd.oipf.ContentAccessDownload+xml"). In that case, the method returns a download identifier. The OITF will then fetch the content access download descriptor, after which the same shall happen as if method registerDownload(), as defined in 4.7.4.2, with the given content access download descriptor as argument was called.

#### 4.7.4.4 Using the optional registerDownloadFromCRID() method

The registerDownloadFromCRID() method is an optional method as defined in 7.4.3 and takes a CRID as one of its arguments that is passed to the underlying native download manager in order to trigger a download.

#### 4.7.4.5 General behaviour regarding triggering a download

The following are general behavioural requirements apply to triggering downloads.

Fetching the content will typically be initiated immediately. However, the OITF may defer the download to a later time.

An OITF should offer an easy way to continue the UI interaction with the server from which a download has been initiated, e.g. allowing him/her to continue browsing on the page that triggered the download.

An OITF should inform the user if the content-type of a content item being retrieved cannot be interpreted by the OITF.

#### 4.7.5 Download protocol(s)

The OITF shall support the HTTP protocol for download as specified in 6.3.4 of IEC 62766-4-1:2017. In addition, the OITF shall support the following requirements.

As specified in 6.3.4 of IEC 62766-4-1, if a server offers a content item for download using HTTP, the server shall make sure that HTTP Range requests as defined in IETF RFC 2616 are supported for HTTP GET requests to the URI of that downloadable content item, in order to be able to resume downloads (e.g. after power or network failure).

If the OITF receives an HTTP 404 "File Not Found" status code, the OITF shall stop its attempts to resume the download, and go to a "Failed Download" state. The handling of other error codes is implementation dependent.

If, after downloading, a content item the size of the downloaded content item does not match the indicated size parameter, or the value for the optional attribute “MD5Hash” of the given <ContentURL> does not match the hash of the downloaded content, the OITF should remove the downloaded content item.

Integration with download protocols other than HTTP are not specified in this document.

## 4.8 Streaming CoD

### 4.8.1 General

Subclause 4.8 defines the content-on-demand streaming interfaces for both DRM-protected and non-DRM protected content.

### 4.8.2 Unicast streaming

#### 4.8.2.1 General

There are three mechanisms by which a reference to content can be passed from a DAE application to the OITF.

- By setting the data property of a A/V Control object as defined in 7.14 to the reference. The application shall set the type attribute to the MIME type of the content referred to by the value of the data attribute to provide a hint about the expected content type, in order for the browser to instantiate the proper object to play the content.
- By setting the src attribute of a <video> element to the reference.
- By including the reference in the <ContentURL> element of a Content Access Streaming descriptor as defined in 7.14.3 and then setting the data property of an A/V Control object as defined in 7.14 to be a reference to that Content Access Streaming Descriptor. In this case the application shall set the type attribute to "application/vnd.oipf.ContentAccessStreaming+xml".

EXAMPLE `<object id="d1" data=http://www.openiptv.org/fetch?contentID=25 type="application/vnd.oipf.ContentAccessStreaming+xml" width="200" height="100"/>`

There are five different possible formats for a reference to unicast streaming content:

- a Public Service Identifier (PSI) as defined in Protocol Specification IEC 62766-4-1;
- an HTTP URL directly referencing the content to be streamed;
- an RTSP URL directly referencing the content to be streamed;
- an HTTP or HTTPS URL referencing a HAS MPD;
- an HTTP or HTTPS URL referencing a MPEG DASH MPD.

All of the mechanisms that an OITF supports shall be supported with all formats of a reference that an OITF supports.

#### 4.8.2.2 HTTP Adaptive Streaming

If the OITF supports HAS content, then it shall support the MIME type as specified for the Media Presentation Descriptor (MPD) in 3GPPTS 26.234, i.e. "video/vnd.3gpp.mpd", and in the HAS specification IEC 62766-2-2. If the OITF supports MPEG DASH content, then it shall support the MIME type as specified for the Media Presentation Descriptor (MPD) in Annex C of ISO/IEC 23009-1:2014, i.e. "application/dash+xml".

The MPD shall be retrieved by specifying a URL. To this end, the OITF shall fetch the MPD from the URL, after which the MPD shall be interpreted and an initial (set of partial) representation(s) selected.

When the URL is passed to the OITF in the data property of a A/V Control object as defined in 7.14, and either a HAS MPD is not valid according to the XML Schema and semantics as defined in Annex A of IEC 62766-2-2:2016 or an MPEG DASH MPD is not valid according to the XML Schema and semantics as defined in ISO/IEC 23009-1 or IEC 62766-2-2, then the A/V Control object shall go to play state 6 ('error'), with error value 4 ('content corrupt or invalid').

If the OITF supports HAS content, then HAS shall also be supported through the video/broadcast object for live content. If the OITF supports MPEG DASH content, then MPEG DASH shall also be supported through the video/broadcast object for live content. This shall be done using Channel objects returned from calls to the createChannelObject( Integer idType, Integer onid, Integer tsid, Integer sid, Integer sourceID, String ipBroadcastID ) method where the idType argument is ID\_IPTV\_URI and the ipBroadcastID argument is a URL which points to an MPD for Scheduled Content (live streaming) over HTTP.

#### 4.8.2.3 Multicast streaming

If an OITF has indicated support for IPTV channels through a <video\_broadcast> element with type ID\_IPTV\_\* (as defined in 7.13.12.2), the OITF shall support passing a content-access descriptor through the 'contentAccessDescriptorURL' argument of the 'setChannel'-method of the video/broadcast' object (as defined in 7.13.2.4). If the content-access descriptor includes DRM information, the OITF shall pass this information to the DRM agent.

### 4.9 Scheduled content

#### 4.9.1 General

If an OITF has indicated support for playback and control of scheduled content through the <video\_broadcast> element, then it shall support the "video/broadcast" embedded object defined in 7.13.2. In addition, it shall adhere to the requirements for conveyance of the channel list as specified in 4.9.2. To protect against unauthorized access to the tuner functionality and people's personal favourite lists, the OITF shall adhere to the security model requirements as specified in 10.1, in particular the tuner-related security requirements in 10.1.4.2.

NOTE 1 This subclause and subclause 7.13 are focused on control and display of scheduled content received over local tuner functionality available to an OITF. The term "tuner" is used here to identify a piece of functionality to enable switching between different types of scheduled content services that are identified through logical channels. This includes IP broadcast channels (using the mechanisms for Scheduled Content defined in IEC 62766-4-1), as well as traditional broadcast channels received over a hybrid tuner.

NOTE 2 The APIs in this subclause allow for deployments whereby the channel line-up and favourite lists for broadcasted content are managed by the client, the server, or a mixture thereof.

#### 4.9.2 Conveyance of channel list

##### 4.9.2.1 General

To enable a service to control the tuner functionality on an OITF, the OITF needs to convey the channel list information that is managed by native code on the OITF device to the service (either the channel list information is provided locally on the OITF via JavaScript, or the channel list is communicated directly to a server). This information includes the list of uniquely identifiable channels that can be received by the physical tuner of a hybrid device, including information about how the channels are ordered and whether or not these channels are part of zero or more favourite lists. It also includes the channel line-up and the favourite lists that may be managed by an OITF for IP broadcast channels.

The API supports two methods of conveying the channel list information to a service:

- a) method 1: through JavaScript, by using the method "getChannelConfig()", as defined in 4.9.2.2;

- b) method 2: through an HTTP POST message that is sent upon the first connection to a service that requires tuner control, as defined in 4.9.2.3.

An OITF shall support method 1, and should support method 2.

If an OITF conveys the channel list information using the HTTP POST message defined in method 2, then the server shall, if it supports method 2, receive the conveyed channel list information and should rely on this information for the purpose of exerting tuner control. If a service supports using the channel list information sent through the HTTP POST method to exert tuner control, the server shall indicate this compatibility with method 2 using the `postList` attribute specified in 9.3.2 (i.e., `<video_broadcast postList="true">true</video_broadcast>`) in the server capability description.

If the server does not support method 2, the service shall rely on the `getChannelConfig()` method defined in 7.13.2.4 to access the channel list information. If an OITF does not support method 2, the HTTP message of the first connection to the service that requires tuner control shall be an HTTP GET message with an empty payload and the service shall instead rely on the `getChannelConfig()` method defined in 7.13.2.4 to access the channel list information. If support for method 2 is indicated by both the OITF and the server (through respective capability exchanges), the OITF shall convey the channel list information using method 2.

If an OITF does not manage/maintain the channel line-up (i.e. does not have a locally stored channel line-up), the `getChannelConfig()` method described in 7.13.2.4 shall return `null`, and the HTTP message described in 4.9.2.3 shall be an HTTP GET message with an empty payload. In that case, the application may use the `createChannelObject()` method as defined in 7.13.2.4 to create channel objects that can be used on subsequent `setChannel()` requests, and in this way can manage/maintain its own channel list.

NOTE Conveyance of the channel list shall adhere to the security model requirements as specified in 10.1.4.2 and 10.1.4.2.2.

#### **4.9.2.2 Method 1: JavaScript method "getChannelConfig()"**

The OITF shall support method "getChannelConfig()" as defined in 7.13.2.4 for the `video/broadcast` embedded object. This method returns a `ChannelConfig` object as defined in 7.13.10.

#### **4.9.2.3 Method 2: HTTP POST message**

If an OITF supports sending the channel list through HTTP POST and a server has indicated that it uses the posted channel list information to exert control of the tuner functionality of an OITF (i.e. using attribute `postList="true"` in the server capability description) for a particular service, then the OITF shall issue an HTTP POST over TLS if it decides to connect to that service. The body of the HTTP POST over TLS request shall contain the client channel listing, which shall adhere to the semantics, syntax and XML Schema that are defined for the client channel listing in Annex G. The server shall silently ignore unknown elements and attributes that are part of the client channel listing.

The server shall return a HTML document.

If the favourite lists are not (partially) managed by the OITF, the client channel listing shall neither contain the "FavouriteLists" nor the "CurrentFavouriteList" element.

#### **4.9.3 Conveyance of channel list and list of scheduled recordings**

This subclause shall apply to OITFs that have indicated `<recording>true</recording>` as defined in 9.3.4 in their capability profile.

To enable a service to schedule recordings of content that is to be broadcast on specific channels, the OITF needs to convey the channel list information that is managed by the native

code on the OITF. This information typically includes the channel line-up of the tuner of a hybrid device. The conveyance of channel list information and scheduled recordings is based on the same two methods of conveying the channel list information to a service as defined in 4.9.2:

- 1) method 1: through JavaScript, by using the method "getChannelConfig()". To this end, the OITF shall support method "getChannelConfig()" as defined in 7.10.2.2 for the application/oipfRecordingScheduler object.
- 2) method 2: through an HTTP POST message as defined in 4.9.2.3 that is sent upon the first connection to a service that has indicated that it requires control of the recording functionality and that has indicated compatibility with method 2 using the postList attribute specified in 9.3.4 (i.e., <recording postList="true">true</recording>), in the server capability description for a particular service.

An OITF shall support method 1, and should support method 2. If support for method 2 is indicated by both the OITF and the server (through respective capability exchanges), the OITF shall convey the channel list information using method 2. Otherwise, the HTTP message of the first connection to the service that requires tuner control shall be an HTTP GET message with an empty payload.

If a server has indicated that it requires control of both the tuner functionality and the recording functionality available to an OITF (i.e. by including both <video\_broadcast> and <recording> with value true in the OITF's capability description), the body of the HTTP POST message shall contain a single instance of the client channel listing whereby the <Recordable> element defined in Annex G shall be used to indicate whether channels that can be received by the tuner of the OITF can be recorded or not.

If an OITF does not manage the channel line-up, the getChannelConfig() method described in 7.10.2.2 shall return null, and the HTTP message described in 4.9.2.3 shall be an HTTP GET message with an empty payload.

In addition, the OITF shall also support method getScheduledRecordings() as defined in 7.10.2.2. This method returns a ScheduledRecordingCollection object, which is defined in 7.10.4.

Note that the conveyance of the channel listing and the scheduled recordings is subject to the security model requirements specified in 10.1, and in particular the recording-related security requirements in 10.1.4.3.

## 4.10 DLNA RUI remote control function

### 4.10.1 General

Subclause 4.10 describes the DLNA RUI RCF (remote control function) and the interactions between the different entities involved. It builds on the RUI feature defined by the IEC 62481 series, the DLNA Networked Device Interoperability Guidelines (August 2009), and shows how the DLNA RUI can be integrated into an OITF and used by DAE applications.

The DLNA RUI RCF is the feature that enables a remote control device to be able to control the OITF or a DAE application running on it, from that remote control device. To support this feature, a remote control device shall support the DLNA RUIC function and an OITF shall support the DLNA RUIS function (as defined in 7.17).

The DLNA RUI RCF provides two main features:

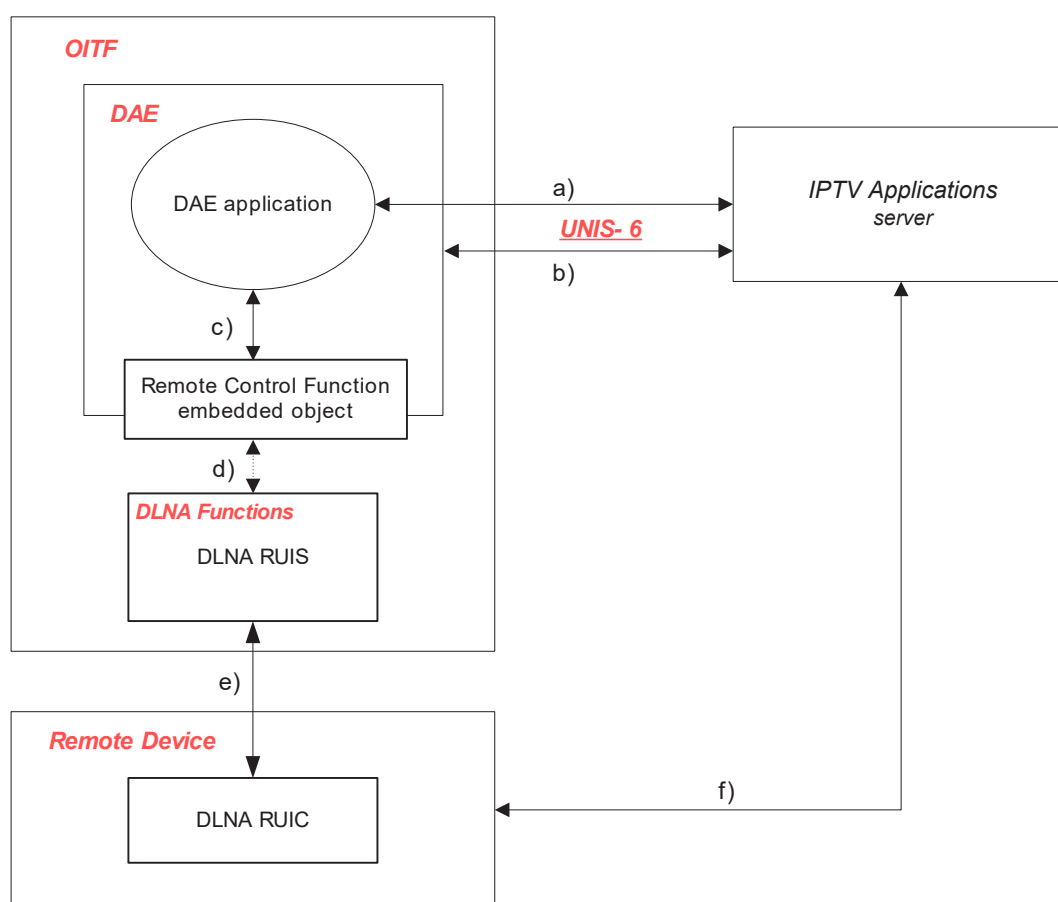
- providing a Control UI to the remote control device:
  - the Control UI is a CE-HTML document through which the user will control the OITF directly or a DAE application on the OITF. There are two options based on the origin of the Control UI for sourcing it as follows:

- sourcing the Control UI from the OITF itself;
- sourcing the Control UI from an IPTV Applications server via the OITF.
- interactions to exchange control messages and results:
  - the Control UI in the DLNA RUIC sends control messages to the OITF or DAE application and receives the corresponding results.

Subclause 4.10.2 introduces the interfaces between the entities that support the DLNA RUI RCF.

#### 4.10.2 Interfaces used by the DLNA RUI remote control function

This subclause describes interfaces related to the DLNA RUI RCF. There are three entities (remote control device, OITF and IPTV applications server) that communicate with each other through the interfaces described in Figure 2.



IEC

Please see 4.10.2 for a description of the interfaces a) to f).

**Figure 2 – OIPF architecture with DLNA RUI RCF scenario**

Figure 2 shows the entities in the OIPF architecture involved in the DLNA RCF and the interfaces between them.

The dotted line "d)" between the RCF embedded object and the DLNA RUIS indicates that it is a local interface and hence not defined by this document. The detailed behaviour of each interface is defined as follows:

Interface a)

This interface is used to retrieve a control UI from an IPTV applications server by using XMLHttpRequest object (the Control UI retrieved through interface a) will be delivered to DLNA RUIC via interfaces c), d) and e), sequentially).

#### Interface b)

This interface is used by the DAE browser to retrieve a DAE application containing an RCF object when the DLNA RUIC requests a DAE application to execute in the OITF.

#### Interface c)

The DLNA RUI RCF APIs use this interface to enable a DAE application to get the request originating from the DLNA RUIC through an event dispatched by the OITF, and send the corresponding response or any other information to the DLNA RUIC via the DLNA RUIS.

#### Interface d)

This is a local interface that is used to pass messages between an RCF object in a DAE application and the DLNA RUIS.

#### Interface e)

This is a DLNA RUI compatible interface which provides device discovery, sending/receiving HTTP messages and notifications.

When the DLNA RUIC is activated by a user, the DLNA RUIC searches for a DLNA RUIS and does a capability exchange. Then, the DLNA RUIC retrieves the XML UI Listing from the DLNA RUIS and displays it to the user. When the user chooses one of the Control UIs, the DLNA RUIC retrieves the selected Control UI from the DLNA RUIS in the OITF.

The Control UI may send an HTTP request to deliver a message (for example, plays an AV content) and receive a response from the DLNA RUIS.

This interface is also used for the DLNA RUIS to send a 3rd party notification defined in 5.6.1 of CEA-2014-A.

#### Interface f)

This interface is used by the selected Control UI (CE-HTML document) to retrieve resources (for example, images, CE-HTML documents, or CSS or JavaScript files) directly from the IPTV Applications server.

## 4.11 Power consumption

### 4.11.1 General

The power states described in this subclause relate to states exposed to the DAE application. There may be other states supported by the OITF, which are not described here.

The OITF will be in one of a number of power states. Its default state is "off", which consumes no power. The OITF shall support an "on" state where it is running in normal operation. The OITF shall support at least one standby state where nothing is being output to the display but power is consumed. An OITF may support two different standby states, "active standby" and "passive standby". An OITF in the "passive standby" state has the smallest possible power consumption (for example, average under 1W), which may be in line with European Commission Code of Conduct, US Energy Star or other regional requirements. In this state, the IR listener and wakeup clock may be active but no DAE application is active. The

listener allows the user to turn on the OITF using a remote control. A DAE application may use the wakeup clock to schedule the OITF enter the "active standby" state, for example to perform a recording.

Note there may be different levels of "active standby" state, but the assumption is that, at least, nothing is being output to the display and one or more DAE applications may execute in the background.

The following explanation describes the behaviour of the OITF when transitioning between the mentioned states and how a DAE application is affected.

A DAE application shall be able to execute in the "on" and "active standby" states but shall not be able to execute in the "off" or "passive standby" states.

When an OITF is turned "on" from an "off" state, a DAE application has to be explicitly selected by the user to be executed or the OITF has identified a DAE application to be auto-started. A DAE application has no direct control if it shall auto-start or not and this is left for the OITF to manage. A DAE application may auto-start if the Service Discovery and Selection has taken place and the user has selected a service provider.

When an OITF changes to an "off" or "passive standby" state from an "on" or "active standby" state, the DAE application shall get an `ApplicationDestroyRequest` event. The DAE application has an opportunity to take a final action and gracefully quit or it shall be killed forcibly.

#### **4.11.2 DAE application wake-up support**

##### **4.11.2.1 General**

The OITF may support wake-up requests from a "passive standby" state. There are two types of wake-up requests, one on an individual DAE application and one on the OITF. The supported wakeup is indicated in the power consumption capability information.

##### **4.11.2.2 Single DAE application wakeup**

The OITF may support wake-up requests for individual DAE applications when in "passive standby". Similar to a scheduled recording, a DAE application may need to execute at a predetermined time. At the wake-up point, the DAE application executes, and when it completes its task returns to a "passive standby" state by exiting.

There shall only be one wake-up request per DAE application. There may be multiple wake-up requests from different DAE applications which shall execute independently. The OITF shall silently ignore all wake-up requests whose timers expire when it is not in the "passive standby" state.

When the DAE application terminates and the OITF changes to an "active standby" or "on" state for other reasons than a wake-up request, the OITF shall not change power states.

Through capability information it is possible to determine if wake-up and standby modes are supported by OITF.

This is an example of how a DAE application may setup a wake-up request in OITF.

Precondition: the DAE application is actively running and the OITF is either in "on" or "active standby" states.

End user selects to go into "passive standby" natively.

An `ApplicationDestroyRequest` event is generated

The DAE application calls the `preparewakeupApplication()` method and sets a token, time for wake-up and URI associated with the DAE application. The DAE application then quits, e.g. by calling `destroyApplication()` on its parent `Application` object.

The OITF goes into "passive standby" state.

When the wake-up time triggers, the OITF changes to "active standby" and the DAE application is initiated with the URI specified in the prior call to `preparewakeupApplication()`.

The DAE application then runs `clearwakeupToken()` to get the token set in the prior call to `preparewakeupApplication()`.

The DAE application executes.

Once the DAE application completes execution, it shall exit. The OITF changes automatically to a "passive standby" state.

If the OITF is turned "on" while in this mode, the OITF shall not enter "passive standby" state.

#### **4.11.2.3 OITF wakeup**

The OITF may support wake-up requests for the OITF when in "passive standby". The application when receiving an event `onApplicationRequest` may request to wake-up the OITF at a set time using method `peparewakeupOITF()`.

OITF shall silently ignore all wake-up requests whose timers expire when it is not in the "passive standby" state.

#### **4.11.3 OITF hibernate support**

The OITF may support a hibernate mode that allows DAE applications and their state to be stored in memory when in a "passive standby" state. The support of a hibernate mode greatly reduces the start-up time for DAE applications (for example, start-up times of 3 s may be reached).

When the OITF resumes from the hibernate mode, it shall restore all of the previous DAE applications with their previous states and should assign the same resources to the DAE applications as they had prior to the hibernate mode. If this is not possible, the regular callback functions shall be used to inform the affected DAE application.

If hibernate mode is supported, the event `ApplicationHibernateRequest` is generated instead of `ApplicationDestroyRequest` when the OITF enters a "passive standby" state.

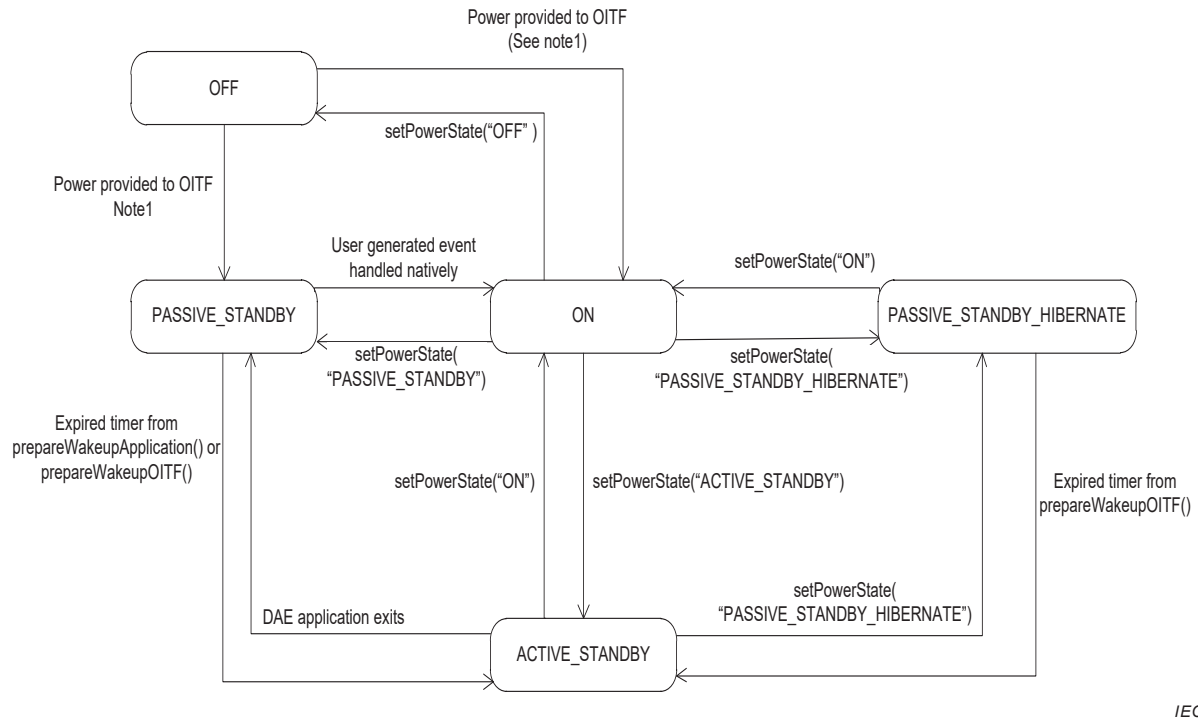
If the OITF supports hibernate mode only, the OITF wake-up request is supported. The single DAE application wake-up shall not be supported. The reason for this limitation is due to the difficulty to support both options.

A wake-up support shall not make the OITF resume from the hibernate mode. The wake-up support shall be supported independently.

The OITF shall indicate support for hibernate mode through the `<hibernateMode>` capability defined in 9.3.20.

#### 4.11.4 State diagram for the power state

The state machine in Figure 3 provides an overview of the power state changes that may occur relating to power consumption. The transitions in the state machine due to `setPowerState()` may be also be triggered by user-generated events handled natively by the OITF.



IEC

NOTE The transition from the OFF state to the PASSIVE\_STANDBY or ON states is manufacturer-dependent.

**Figure 3 – State diagram of OITF power states**

#### 4.12 Display model

Annex H describes the logical display model of an OITF and the relationship between DAE application graphics and video.

### 5 DAE application model

#### 5.1 Application lifecycle

##### 5.1.1 General

Subclause 5.1 describes the lifecycle of a DAE application, including when an application is launched, when it is terminated and the behaviour when a DAE leaves the boundary of one application and enters another.

APIs related to DAE applications are described in 7.2.

##### 5.1.2 Creating a new application

###### 5.1.2.1 General

The present list defines a number of different application lifecycle models. These include:

- applications started through an OITF-specific user interface;

- using the `Application.createApplication()` API call;
- CE-HTML third party notifications;
- service provider related applications (from SD&S signalling);
- applications started by the DRM agent;
- applications provided by the AG through the remote UI;
- broadcast-related applications (either be from SD&S signalling or from broadcast signalling in a hybrid device);
- broadcast-independent applications;
- widgets.

#### 5.1.2.2 Broadcast-independent applications

Broadcast-independent applications are started by fetching the first page of the application from a URL.

#### 5.1.2.3 Applications started through an OITF-specific user interface

These shall be presented as broadcast-independent applications.

#### 5.1.2.4 Using the `Application.createApplication()` method

Creating a new application is accomplished by creating a new `Application` object via the `Application.createApplication()` method. Calling this method will create a new application and add it to the application tree in the appropriate location.

```
// Assumes that the application/oipfApplicationManager object has the ID
// "applicationmanager"
var appMgr = document.getElementById("applicationmanager");
var self = appMgr.getOwnerApplication(window.document);

// create the application as a child of the current application
var child = self.createApplication( url_of_application, true );
```

The URL passed to the `createApplication()` method shall be one of the following:

- an HTTP or HTTPS URL referring to an XHTML page as defined by 6.1;
- an HTTP or HTTPS URL referring to an XML AIT as defined by 5.2.7.1;
- the DVB URI for launching service provider related applications signalled through SD&S as defined in 8.3;
- the DVB URI for launching broadcast-related applications from the current service signalled through SD&S as defined in 8.3. Where an OITF supports the MPEG-2 encoding of the AIT as defined in 5.2.7.2, this form of the DVB URI shall also be supported for launching broadcast-related applications from the current service when that service includes an MPEG-2 AIT.

#### 5.1.2.5 CE-HTML third party notifications

The lifecycle of these is defined in CEA-2014-A and summarised in 5.3.2.

#### 5.1.2.6 Starting applications from SD&S Signalling

These are described in 5.2. All applications started by SD&S signalling are treated as siblings and are children of the hidden system root node (see 4.4.4).

#### 5.1.2.7 Applications started by the DRM agent

These shall be considered as broadcast-independent applications.

#### 5.1.2.8 Applications provided by the AG through the remote UI

OITFs may include the capability to start these applications from an embedded application. OITFs shall include the ability for applications to discover these as defined by the "application/oipfGatewayInfo" embedded object in 7.7.2.

#### 5.1.3 Stopping an application

The `destroyApplication()` method (as specified in 7.2.3.3) shall terminate the application. An application may register a listener on the `ApplicationDestroyRequest` event in order to perform any clean-up before being destroyed completely. After the `destroyApplication()` method returns, further execution of the specified application shall not occur.

When an application is terminated, all associated resources shall be freed (or marked available for garbage collection). Any active network-related sessions will be terminated. Any media content being presented by the application is stopped, although recordings or content downloads initiated by the application will not be affected.

NOTE Terminating an application does not imply any effect on the state of the DAE environment.

Additional requirements are defined for stopping selected service provider applications and applications that are part of scheduled content services in 5.2.4.3 and 5.2.3.2 respectively.

#### 5.1.4 Application boundaries

All of the pages that make up an application are contained within its application boundary. This is the "fully qualified domain name" (FQDN) of the initial page of the application in the absence of a `simple_application_boundary_descriptor`.

If an `applicationBoundary` element is present in the SD&S signalling for an application as defined in ETSI TS 102 809, the application boundary shall also include the FQDNs listed in the `applicationBoundary` element. If this element is not present, then the application boundary shall consist of the FQDN of the initial page of the application.

For files requested with `XMLHttpRequest`, the same origin policy shall be extended using the application domain; i.e. any domain in the application domain shall be considered of same origin.

The OITF shall remove any IP address in the application boundary which is within the private address space as defined in IETF RFC 1918, before launching the application.

Extending the origin of `XMLHttpRequest` is potentially dangerous, and may lead to undesired leaking of private information. To make sure that the integrity of the user is not compromised, the OITF should include a mechanism which allows the user to exclude domains from application boundaries of applications.

### 5.2 Application announcement and signalling

#### 5.2.1 Overview

This document defines 3 basic types of application:

- applications related to one or more broadcast TV or radio channels. These may run while one of the channels which they are related to is being presented by the OITF. These are signalled through the SD&S broadcast or package discovery records or included in an application discovery record which is referenced from the broadcast or package discovery record.
- applications related to the service provider selected through the service selection process. These may run at any time until the service provider selection process is repeated and a

different service provider selected. These are signalled through the SD&S service provider discovery record or included in an application discovery record which is referenced from the service provider discovery record.

- applications independent of either of the above. These may run at any time. These are started by other applications and are not signalled anywhere.

Each of these types is described in more detail in 5.2.2 to 5.2.8.

### 5.2.2 General

Subclause 4.4.4 describes how one application may start another application either as a sibling or as a child. All applications started via SD&S signalling as described in this subclause shall be started as children of the hidden system root node, as described in 5.1.2.6.

Any application may be signalled as AUTOSTART or PRESENT (see Table 3 and 5.2.4.3 of ETSI TS 102 809). Applications signalled as AUTOSTART are intended to be automatically started by the OITF. Applications signalled as PRESENT are intended to be started only by other applications. Broadcast-related applications may alternatively be signalled as KILL or PREFETCH.

It is up to the OITF manufacturer to ensure a good quality of experience concerning:

- navigation within a DAE application;
- accessing the available DAE applications, both available for launch, and those already running;
- managing the life cycles of all DAE applications able to be used concurrently.

It is outside the scope of this document whether there are dedicated keys on a remote control (e.g. the "menu", "home" or "guide" key), there is an entry in an on-screen menu or there are some other mechanism.

It is optional for the OITF to support an exit mechanism directly accessible by the end-user. If one is supported, it is outside the scope of this document whether this mechanism is a button on a remote control, an item in an on-screen menu or something else. If such a mechanism is supported then it shall only stop the application the end-user is currently interacting with and any child applications of that application. The parent application and any siblings shall not be stopped.

Additionally, any application may be stopped under the following circumstances:

- the application itself exits;
- its parent application exits;
- it is stopped by the application which started it or another application which has a reference to its application object;
- in response to changes in the application signalling as defined below for broadcast-related applications and service provider related applications.

In all these above cases, except the first (when an application itself exits), when an application is stopped by the OITF, an `ApplicationDestroyRequest` event (as defined in 7.2.7) shall be raised on the application. In the following error conditions, an application being stopped should have an `ApplicationDestroyRequest` event raised if this is possible:

- the OITF runs out of resources for applications and has to stop some of them in order to keep operating correctly;
- the OITF has determined that an application is non-responsive or has crashed.

### 5.2.3 Broadcast-related applications

#### 5.2.3.1 General

Providers of broadcast TV channels may signal broadcast-related applications as part of the SD&S broadcast discovery record (see 4.3.3 of IEC 62766-3:2016, as well as 4.2.1 and 5.4.3.2 of ETSI TS 102 809). As an optimisation, broadcast-related applications that are associated with a group of channels may be signalled as part of the SD&S package discovery record (see 4.3.3 of IEC 62766-3:2016, as well as subclause 5.4.3.1 of ETSI TS 102 809). Broadcast-related applications may be included in the SD&S broadcast discovery or package discovery records or included in an application discovery record that is referenced from the broadcast discovery record.

Broadcast-related applications can also be signalled in-line in an MPEG-2 transport stream using the MPEG-2 encoding of the AIT as defined in 5.2.7.2 below.

When a broadcast TV channel starts being presented, the OITF shall follow the procedure defined in 5.2.3.3 below.

While a broadcast TV channel is being presented, the OITF shall monitor for changes in the SD&S information as defined by 5.2.2.3 of IEC 62766-3:2016.

When changes are detected, the OITF shall follow the procedure defined in 5.2.3.4.

**NOTE** The typical "red button" behaviour can be achieved by having the first page of an AUTOSTART broadcast related application be full screen and transparent to video except for an image showing a red button. Only when the user generates a "red" key event does the application display more of its user interface.

OITFs may include the capability to start and stop a broadcast-related DAE application instead of analogue teletext services as part of a scheduled content service or channel. Typically, this would re-purpose the same mechanism used to start an analogue teletext service – for example a "text" button on a remote control. These are identified using the application usage mechanism defined in ETSI TS 102 809 and 5.2.7.

#### 5.2.3.2 Stopping

In addition to what is stated in 5.2.2, broadcast related applications are stopped when:

- changing between channels as defined in the procedure in 5.2.3.3;
- the OITF detects an update to the signalling for a currently presented channel as defined in the procedure in 5.2.3.4;
- the OITF stops presenting any broadcast channel.

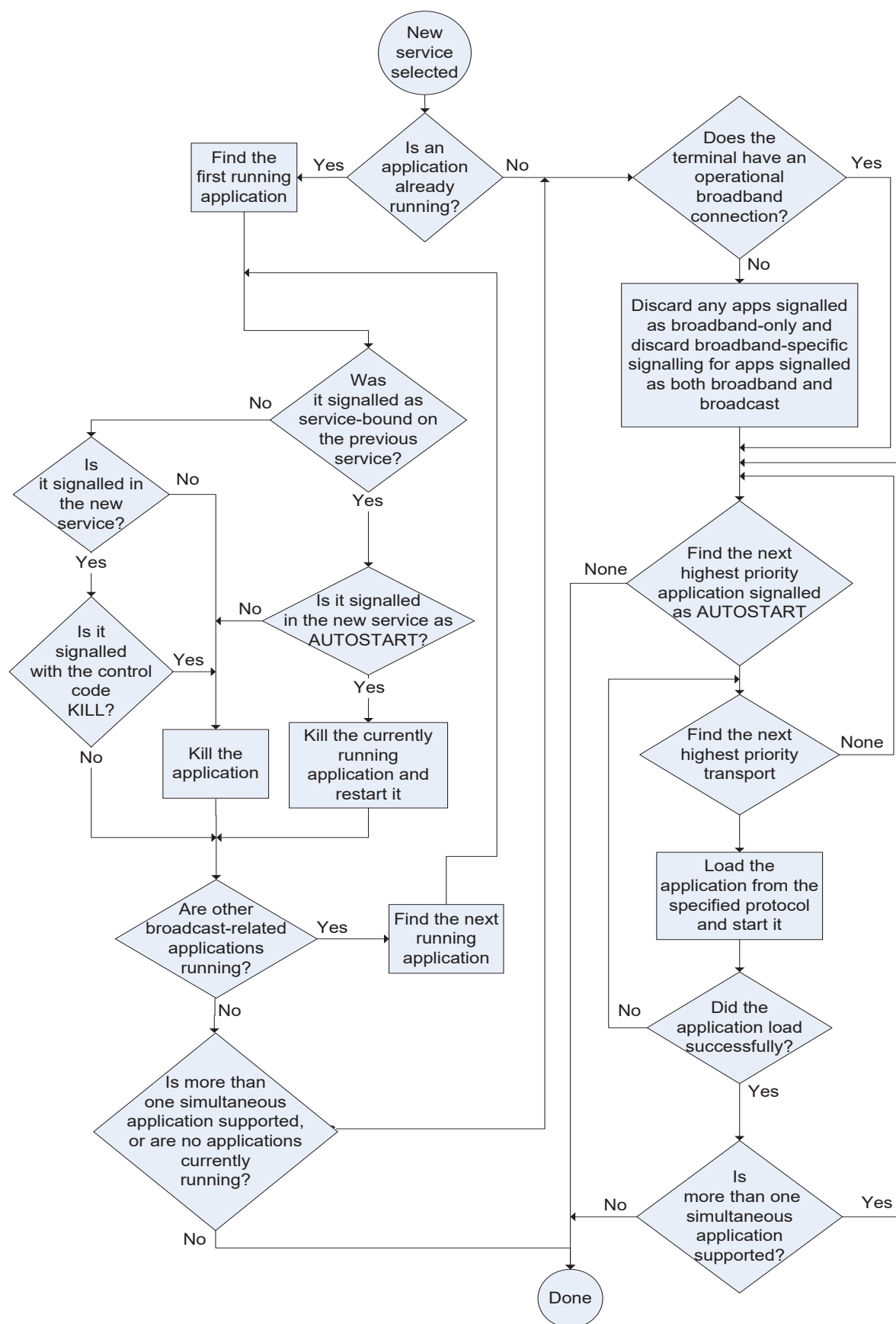
#### 5.2.3.3 Procedure for starting and stopping broadcast related applications on channel change

When a scheduled content service is selected, the following shall apply:

- the OITF shall determine if there are any applications signalled as part of the service as defined by 4.3.4.2 and 4.3.4.3 of IEC 62766-3:2016;
- applications that are related to that scheduled content service and which are signalled with a control code of AUTOSTART shall be started if not still running from any previously presented linear TV service; they shall be started commencing with the highest priority application working downwards in priority while resources in the OITF permit;
- applications that are related to that scheduled content service, which are signalled with a control code of AUTOSTART and that are already running from a previously presented scheduled content service shall:
  - continue to run uninterrupted if the serviceBound element of the ApplicationDescriptor in their signalling has value false;

- be stopped and re-started if the serviceBound element of the ApplicationDescriptor in their signalling has value true.
- applications that are related to that scheduled content service and which are signalled with a control code of PRESENT shall continue to run if already running but shall not be started if not already running;
- running applications from any previously presented scheduled content service that are not part of the new scheduled content service shall be stopped as part of the change of presented service.

Figure 4 shows the behaviour that shall apply when the selected channel changes.



IEC

Figure 4 – Behaviour when the selected channel changes

#### **5.2.3.4 Procedure for starting and stopping broadcast related applications when signalling is updated**

When the application signalling for a scheduled content service is updated, the following apply:

- applications that are added to the service with a control code of AUTOSTART shall be automatically started when their addition is detected by the OITF. They shall be started commencing with the highest priority application working downwards in priority while resources in the OITF permit; applications added to the service with any other control code shall not be automatically started;
- applications that are part of the service whose control code changes to AUTOSTART from some other value shall be automatically started unless already running;
- an application that is removed from the service or whose control code changes to KILL shall be stopped.

If application signalling is removed from a service, all running broadcast-related applications shall be stopped (i.e. the same behaviour as signalling an empty AIT).

Figure 5 shows the behaviour that shall apply when the application signalling for the currently selected service changes, or when a running broadcast-related application exits.