OIPF

Release 2 Specification

Volume 6 – Procedural Application Environment

[V2.2] – [2013-05-22]

Open IPTV Forum

Open IPTV Forum

Postal address

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

Internet
http://www.oipf.tv

Disclaimer

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

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

Copyright Notification

Contents

FOREWORD **7**

INTRODUCTION **7**

1 SCOPE **9**

2 REFERENCES **10**

2.1 Normative References **10**
2.2 Open IPTV Forum References **10**

3 CONVENTIONS AND TERMINOLOGY **11**

3.1 Conventions **11**
3.2 Terminology **11**
3.2.1 Abbreviations 11

4 GENERAL CONSIDERATIONS AND CONVENTIONS **12**

4.1 Relation between DVB-GEM and PAE Definitions **12**
4.1.1 DVB-GEM Compliance 12
4.1.2 Functional Equivalents 13

5 ARCHITECTURE AND DEPLOYMENT OPTIONS **14**

5.1 Architecture **14**
5.2 Deployment Options **15**
5.2.1 Combined IG-AG-OITF STB and OITF TV (“headed configuration”) 15
5.2.2 Combined AG-IG with multiple OITFs (“headless configuration”) 16
5.2.3 AG-IG, OITF-IG, Multiple OITFs 17
5.2.4 Combined OITF-AG TV and IG-WAN Gateway (“headed configuration”) 18
5.3 Remote UI Server (informative) **18**

6 PROTOCOLS **19**

6.1 Broadcast Channel Protocols **19**
6.2 Interaction Channel Protocols **19**
6.3 Transport protocols for application loading over the interaction channel **19**
6.4 IPTV Protocols **19**
6.4.1 Streaming Protocols 19
6.4.2 Metadata Protocols 20
6.4.3 Content Download Protocols 20
6.5 Home Network Protocols **20**

7 CONTENT FORMATS **21**

7.1 Static formats **21**
7.2 Streaming formats **21**
7.3 Fonts **21**
7.3.1 Resident Fonts 21
7.3.2 Downloadable Fonts 21

8 VOID **22**

9 APPLICATION MODEL **23**

9.1 Introduction (informative) **23**
9.2 Broadcast Applications **24**
9.3 DVB-J Model **24**
9.4 Stored and Cached Applications **24**
9.5 Unbound Applications **24**

10 APPLICATION SIGNALLING / METADATA **25**

10.1 XML AIT **25**
10.2 Stored and cached applications **26**

11 THE JAVA PLATFORM **27**

11.1 Fundamentals **27**
11.2 Extensions and mappings to GEM APIs **27**
11.2.1 Broadcast Transport Protocol Access API (org.dvb.dsmcc) 27
11.2.2 Application Listing and Launching API (org.dvb.application) 28

11.2.3 Streaming Media APIs ... 28
11.2.4 GEM 3D API ... 28
11.3 APIs defined by this Volume ... 28
11.3.1 Content and Service Protection API ... 28
11.3.2 User Authentication API ... 28
11.3.3 UI Server API ... 28
11.3.4 Content download API ... 28
11.3.5 Service API ... 28
11.4 PVR APIs ... 28
11.5 Content referencing ... 29
12 SECURITY ... 30
12.1 Authentication of Applications ... 30
12.2 Permission request file ... 30
12.3 Security Policy for Applications ... 30
12.4 Certificate Management ... 30
13 GRAPHICS REFERENCE MODEL ... 31
14 SYSTEM INTEGRATION ASPECTS ... 32
15 DETAILED PROFILE DEFINITIONS ... 33
16 PVR ... 43
16.1 Mandatory Responsibilities ... 43
16.2 Optional Responsibilities ... 44
16.3 Visibility of Recording Requests and Recordings between Applications and Service Providers ... 45
17 MINIMUM TERMINAL CAPABILITIES ... 46
18 HTTP ADAPTIVE STREAMING ... 47
18.1 HAS support ... 47
APPENDIX A. HEADLESS BEHAVIOUR OF UI-RELATED APIS (INFORMATIVE) ... 48
A.1 PBP ... 48
A.2 JavaTV ... 48
APPENDIX B. VOID ... 50
APPENDIX C. PACKAGE ORG.OIPF.DOWNLOAD ... 51
INTERFACE APPLICATIONDOWNLOADREQUEST ... 51
CLASS LOCATORDOWNLOADSPEC ... 51
CLASS APPLICATIONDOWNLOADEXCEPTION ... 52
CLASS APPLICATIONDOWNLOADSPEC ... 53
APPENDIX D. PACKAGE ORG.OIPF.SERVICE ... 54
INTERFACE SERVICECREATOR ... 54
APPENDIX E. ORG.OIPF.AUTH ... 55
CLASS HTTPDIGESTCREDENTIALS ... 55
CLASS USERAUTHENTICATIONPERMISSION ... 55
CLASS USERAUTHENTICATIONMANAGER ... 56
CLASS USERCREDENTIALS ... 57
CLASS COOKIECREDENTIALS ... 57
APPENDIX F. ORG.OIPF.UISERVER ... 59
CLASS UISERVERMANAGER ... 59
APPENDIX G. ORG.OIPF.DRM ... 60
CLASS DRMAGENTEVENT ... 60
CLASS DRMAGENTPERMISSION ... 61

INTERFACE DRMAGENTLISTENER 62
CLASS DRMAGENT 63
CLASS DRMAGENTEXCEPTION 64
CLASS DRMRIGHTSERROREVENT 65
APPENDIX H. ORG.OIPF.PVR 67
CLASS RECORDINGACCESSPERMISSIONS 67
CLASS RECORDINGPROPERTIES 68

Tables

Table 1: Status of XML AIT Descriptors and Elements....25
Table 2: Mapping of GEM Clauses Relating to Content Referencing....29
Table 3: Locators and Corresponding Text Representations....32
Table 4: Platform Profile Definitions in this Volume....33
Table 5: Applicability of GEM Specification Sections....34
Table 6: Summary of Functional Equivalents (Informative)....39
Table 7: Responsibilities of GEM Recording Specifications....43
Table 8: Events During Normal Playback and Resulting Behaviour....44
Table 9: Optional Responsibilities of GEM Recording Specifications....44

Figures

Figure 1: Architecture Block Diagram....14

Foreword

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

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

Introduction

The Open IPTV Forum Release 2 Specification consists of nine Volumes:

- Volume 1 – Overview,
- Volume 2 – Media Formats,
- Volume 2a – HTTP Adaptive Streaming,
- Volume 3 – Content Metadata,
- Volume 4 – Protocols,
- Volume 4a – Examples of IPTV Protocol Sequences,
- Volume 5 – Declarative Application Environment,
- Volume 6 – Procedural Application Environment (this volume), and
- Volume 7 – Authentication, Content Protection and Service Protection.

This volume defines the Procedural Application Environment (PAE) available for Release 2 Open IPTV compliant services and devices. Like other specifications such as OCAP, ACAP and BluRay, which are GEM terminal specifications, this volume follows the structure of the [GEM] specification.

- Chapter 1 describes the scope of the procedural application environment.
- Chapter 2 is References.
- Chapter 3 is Definitions and Abbreviations
- Chapter 4 covers General Considerations and Conventions
- Chapter 5 explains the relationship between DVB-GEM and the PAE platform, the basic architecture and positioning of the procedural application environment in the OIPF landscape.
- Chapter 6 includes details on the supported transport protocols by cross-referencing the OIPF Protocols specification [OIPF_PROT2].
- Chapter 7 defines static and streaming media formats for the representation of images, sound, videos, colors and fonts by cross-referencing the OIPF Media Formats specification [OIPF_MEDIA2].
- Chapter 8 is intentionally void.
- Chapters 9 and 10 describe the Application Model for Java applications for the PAE and application signalling
- Chapter 11 describes features of the Java platform such as class loading behaviour, event model and specifies all APIs that are contained in the platform.

- Chapter 12 deals with security aspects of the platform such as the security framework for applications, application authentication, secure network connections and certificate management.
- Chapter 13 deals with the graphics reference model
- Chapter 14 deals with system integration aspects
- Chapter 15 defines which parts of the specification are mandatory or optional for the 3 different device types addressed by this volume.
- Chapter 16 deals with PVR functionalities
- Chapter 17 defines the Minimum Terminal Capabilities
- Chapter 18 deals with HTTP Adaptive Streaming support.

The Annexes contain API definitions at class and method level and clarifications of referenced specifications.

1 Scope

This volume defines the UNI Reference Point UNIS-12 of the Open IPTV Forum Functional Architecture [OIPF_ARCH2]

2 References

2.1 Normative References

[GEM]	ETSI, TS 102 728 V1.2.1 (2011-09), "Digital Video Broadcasting (DVB); Globally Executable MHP (GEM) Specification 1.3 (including OTT and hybrid broadcast/broadband)"
[GEMS3D]	ETSI, TS 101 600 V1.1.1 (2012-05), "Digital Video Broadcasting (DVB); GEM Profile for Plano-Stereoscopic 3DTV"
[PBP 1.1]	Java Community Process, Java Specification Request JSR-217, "Personal Basis Profile (PBP) 1.1" or later. NOTE: The latest release of JSR 217 is available at http://www.jcp.org/en/jsr/detail?id=217
[CDC 1.1]	Java Community Process, Java Specification Request JSR-218, "Connected Device Configuration (CDC) 1.1" or later. NOTE: The latest release of JSR 218 is available at http://www.jcp.org/en/jsr/detail?id=218
[FP 1.1]	Java Community Process, Java Specification Request JSR-219, "Foundation Profile (FP) 1.1" or later. NOTE: The latest release of JSR 219 is available at http://www.jcp.org/en/jsr/detail?id=219
[JAVA TV]	Java Community Process, Java Specification Request JSR-927, "Java TV, Version 1.1.1" or later. NOTE: The latest release of JSR 927 is available at http://www.jcp.org/en/jsr/detail?id=927
[MHP]	ETSI, TS 102 727 V1.1.1 (2010-01), "Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.2.2"
[RFC2119]	IETF, RFC 2119, "Key words for use in RFCs to indicate requirement levels"
[OCSP]	IETF, RFC 2560 "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP"
[BCG]	ETSI, TS 102 539 V.1.1.1, "Digital Video Broadcasting (DVB); Carriage of Broadband Content Guide (BCG) information over Internet Protocol (IP)"
[JAR]	Java Community Process, "JAR File Specification, part of Java SDK 1.4.2 specification", 1999 http://java.sun.com/j2se/1.4.2/docs/guide/jar/jar.html
[DVR]	ETSI, TS 102 817 V1.1.1, "Digital Video Broadcasting (DVB);Digital Recording Extension to Globally Executable Multimedia Home Platform (GEM)"
[MHP-PVR]	ETSI, TS 102 816 V1.1.1, "Digital Video Broadcasting (DVB); Personal Video Recorder (PVR)/Personal Data Recorder (PDR) Extension to the Multimedia Home Platform"
[HEADLESS]	DVB, Blue Book A127, "Application Gateway and Media Server Fragment". NOTE: http://www.mhp.org/specs/a127.application_gateway_and_media_server_fragment.pdf
[CEA-2014-A]	Consumer Electronics Association, CEA-2014-A, July 2007, "Web-based Protocol Framework for Remote User Interface on UPnP Networks and the Internet (Web4CE)", including the August 28, 2008 Errata.
[TS26234]	3GPP TS 26.234, Transparent end-to-end Packet-switched Streaming Service (PSS) Protocols and codecs (Release 9)

2.2 Open IPTV Forum References

[OIPF_ARCH2]	Open IPTV Forum, "Functional Architecture - V2.2", April 2013.
[OIPF_MEDIA2]	Open IPTV Forum, "Release 2 Specification, Volume 2 - Media Formats", V2.2, April 2013.
[OIPF_HAS2]	Open IPTV Forum, "Release 2 Specification, Volume 2a - HTTP Adaptive Streaming", V2.2, April 2013.
[OIPF_META2]	Open IPTV Forum, "Release 2 Specification, Volume 3 - Content Metadata", V2.2 April 2013.
[OIPF_PROT2]	Open IPTV Forum, "Release 2 Specification, Volume 4 - Protocols", V2.2, April 2013.
[OIPF_DAE2]	Open IPTV Forum, "Release 2 Specification, Volume 5 - Declarative Application Environment", V2.2, April 2013.

3 Conventions and Terminology

3.1 Conventions

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

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

3.2 Terminology

3.2.1 Abbreviations

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

Acronym	Definition
DSMCC	Digital Storage Media Command and Control
FP	Foundation Profile
GEM	Globally Extensable MHP
HAS	HTTP Adaptive Streaming
JSSE	Java Secure Sockets Extension
MHP	Multimedia Home Platform
MPD	Media Presentation Description
PBP	Personal Basis Profile

4 General considerations and conventions

4.1 Relation between DVB-GEM and PAE Definitions

The GEM/MHP notion of “Broadcast Application” SHOULD be interpreted as meaning applications that are bound to a specific scheduled content service or a content on demand item.

Unbound applications are persistent applications that are not tied to any service.

The GEM term “interaction channel” is the network connection channel of the OIPF architecture [OIPF_ARCH2].

4.1.1 DVB-GEM Compliance

All mandatory requirements for the GEM IPTV target SHALL be supported by this document.

The following optional parts of the GEM IPTV target SHALL be supported by this document;

- OpenType
- Internet Access applications
- Stored applications, stored services and corresponding APIs
- Unbound applications and corresponding APIs
- Content referencing for IPTV
- Service discover and selection for IPTV

The following optional parts of the GEM IPTV target SHALL be supported by this document under specific circumstances.

- TV-Anytime content referencing and metadata

The following optional parts of the GEM IPTV target MAY be supported by this document;

- File storage API
- Smart Card API (JSR177)
- Providers
- Privileged applications

Where a service provider is specifying a device including the PAE, the service provider MAY choose to require these optional features.

The following optional parts of the GEM IPTV target SHOULD NOT be supported by this document;

- DVB-HTML
- The MHP functional equivalent called application authentication.
- Section filtering API (org.davic.mpeg.sections), Tuning API, Basic MPEG concepts and common error reporting
- MPEG 2 video drips and corresponding APIs
- MPEG 2 I-frames and corresponding APIs
- Credentials
- Certificate revocation mechanism of [MHP]
- Root certificate management mechanism defined in [MHP]

- Externally authorized applications

4.1.2 Functional Equivalents

The functional equivalents to GEM-IPTV are listed in table “Table: Functional equivalents”.

The main concepts are as follows;

- DVB Service Discovery and Selection and Broadband Content Guide are used for SI.
- Applications are distributed in JAR files, using HTTP for unicast and FLUTE for multicast.
  NOTE: FLUTE is conceptually different from DSMCC object carousel and cannot be considered a functional equivalent.
- Application authentication is based on the signing capabilities for JAR files.

5 Architecture and Deployment Options

5.1 Architecture

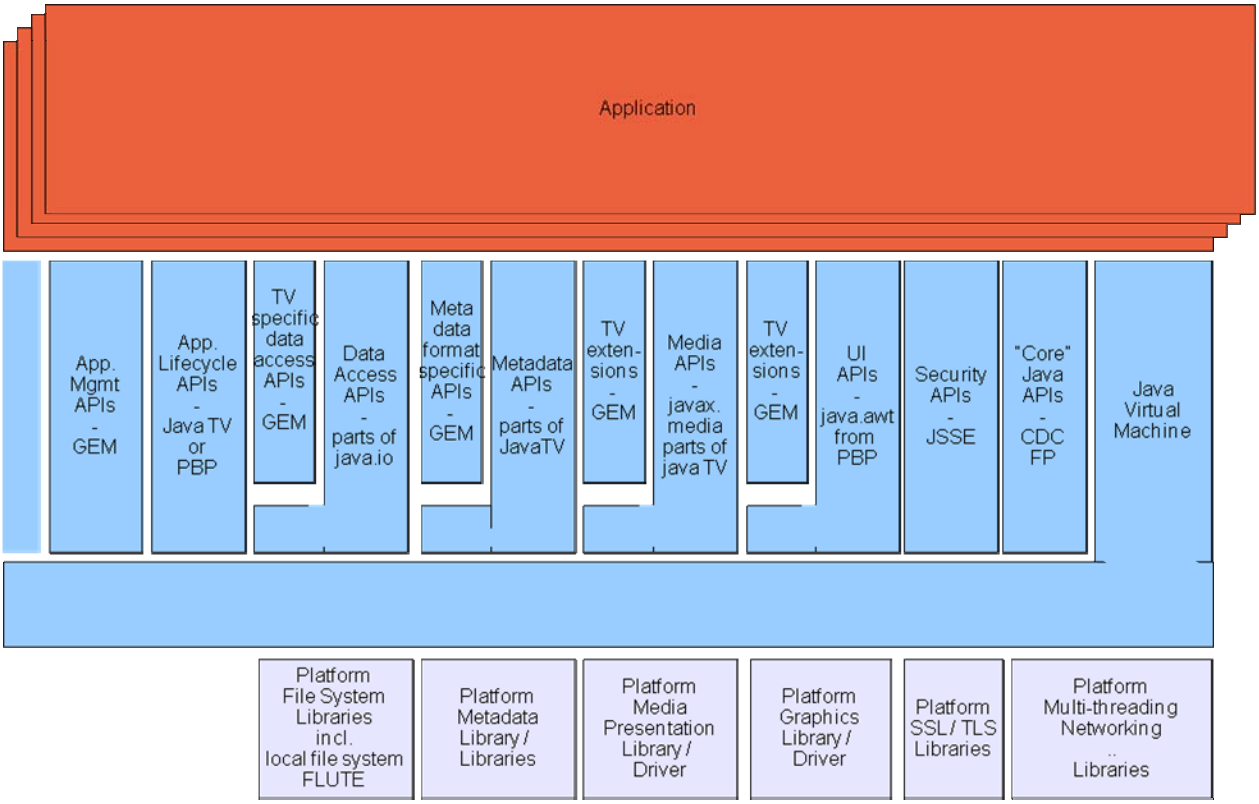


Figure 1: Architecture Block Diagram

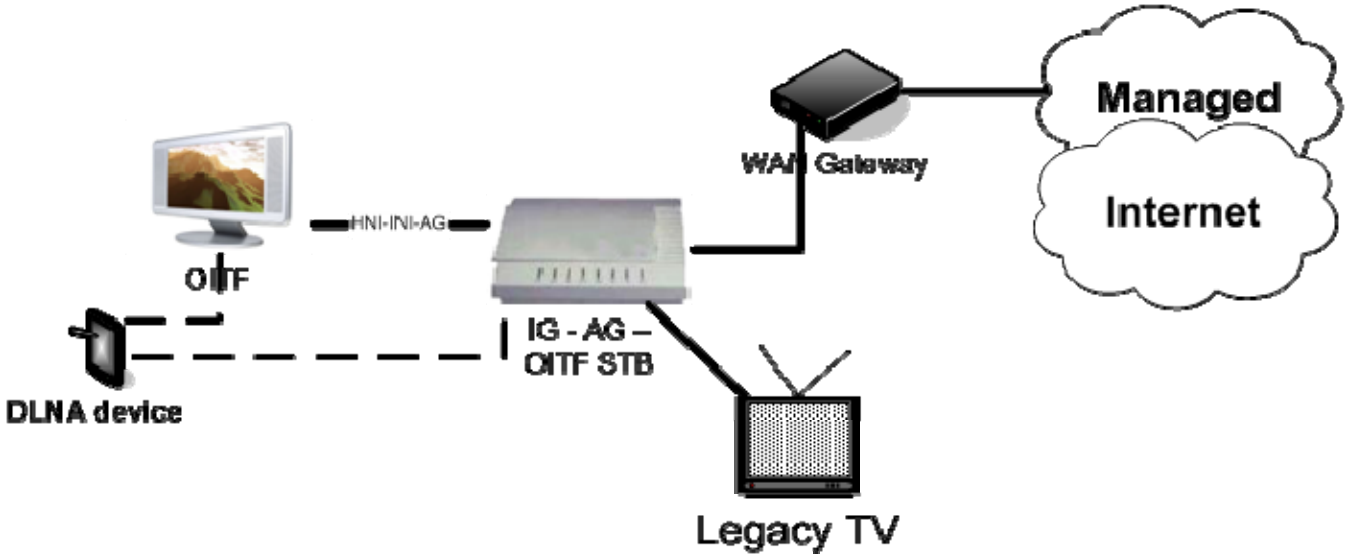
The above figure shows a number of the key components of the architecture and indicates their origin, GEM, JavaTV, PBP or components of PBP (JSSE, CDC or FP). Below is a short summary of each component ordered from left to right in the figure.

- App Mgmt APIs: These enable one application to obtain lists of other applications and start or stop other them.
- App. Lifecycle APIs: These are the first APIs to be called by the implementation when starting an application.
- TV specific data access APIs: These provide access to data in ways which are specific to TV environments
- Data access APIs: These provide generic access to data – files, sockets, streams, etc.
- Metadata format specific APIs: These provide access to the details of metadata in a format dependent way.
- Metadata APIs: These provide access to metadata and are independent of any specific metadata format.
- TV extensions: These provide control over A/V media in ways which are specific to TV environments
- Media APIs: These provide access to and control over A/V media
- TV extensions: These provide UI features specific to TV environments
- UI APIs: These provide the basic UI capabilities
- Security APIs: These provide the basic security capabilities.
- Core Java APIs: These provide the basic capabilities of the Java language.
- Java virtual machine: This provides the basic Java capabilities of the environment.

5.2 Deployment Options

The following deployment options were taken from chapters 5.3.4.2 of [OIPF_ARCH2] for reference purposes.

5.2.1 Combined IG-AG-OITF STB and OITF TV ("headed configuration")

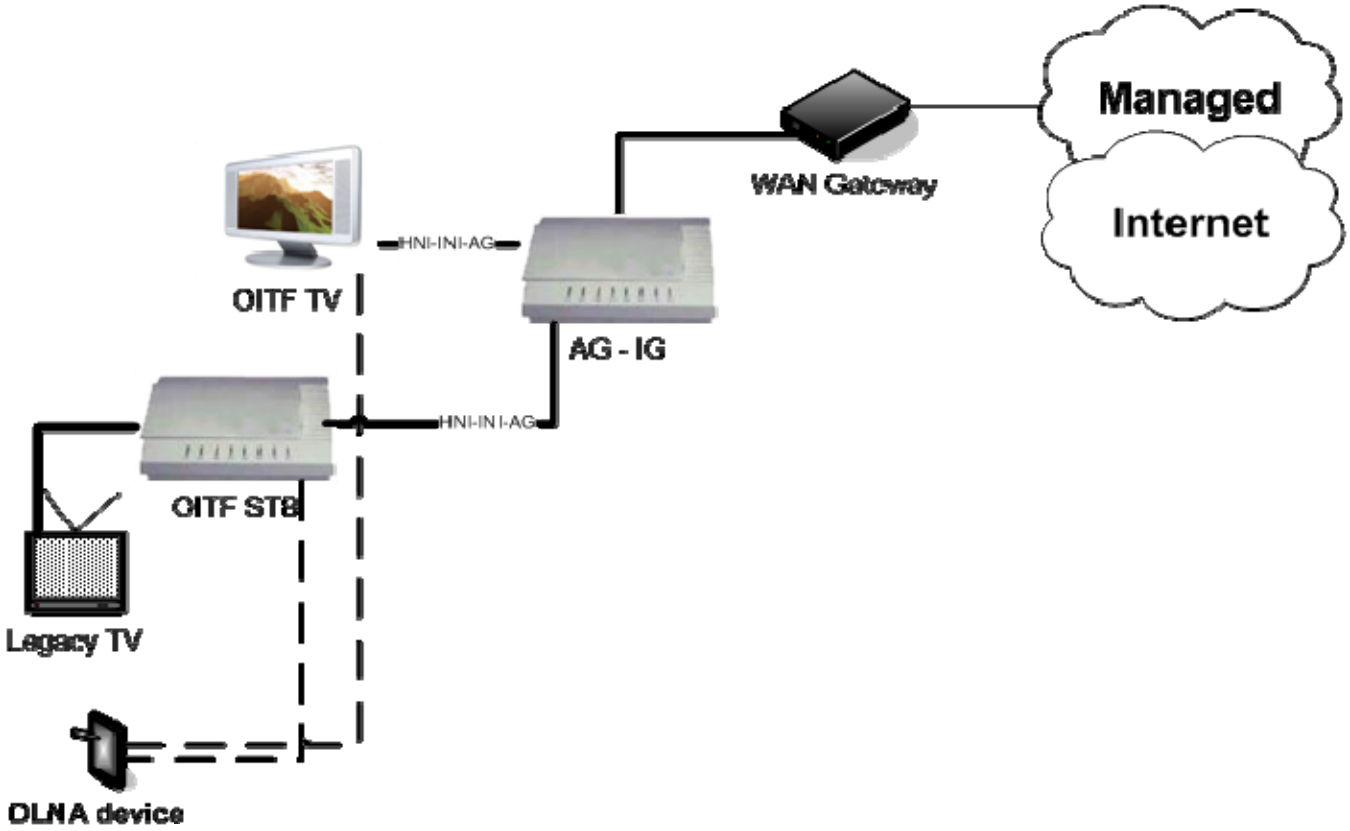


This deployment supports both managed and unmanaged services, with DAE and PAE applications, presented on an OITF TV and a legacy TV. The following devices are deployed:

- A WAN Gateway.
- Combined IG, AG and OITF STB: A set top box including IG, AG and OITF functionality, that exposes HNI-INI-AG to other OITFs in the residential network. It connects to the legacy TV using some non-OIPF specified mechanism, such as HDMI or SCART.
- OITF TV: This is a TV containing an OITF.

Optionally, the IG-AG-OITF STB or the OITF TV may act as a DLNA DMS to make OIPF services available to DLNA devices. They may also act as a DLNA DMP to access content from other DLNA devices on the home network.

5.2.2 Combined AG-IG with multiple OITFs ("headless configuration")

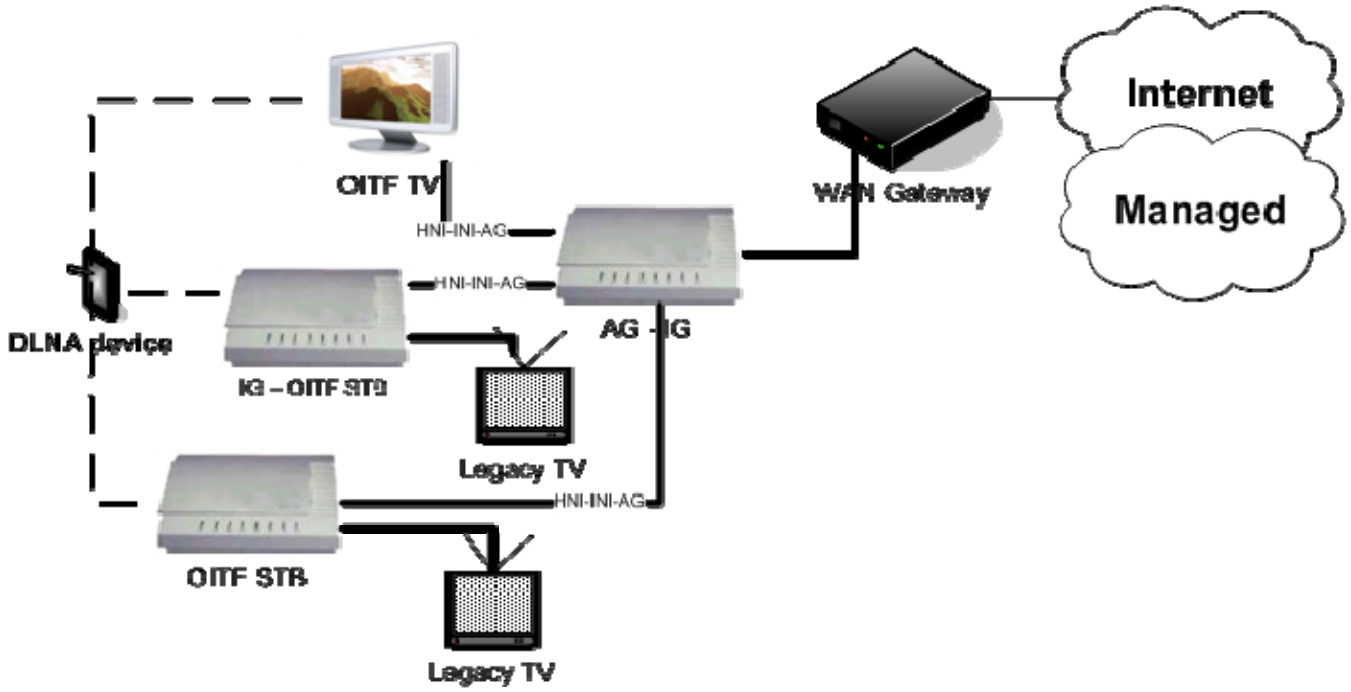


This deployment supports both managed and unmanaged services, with DAE and PAE applications, presented on an OITF TV and a legacy TV. The following devices are deployed:

- A WAN Gateway.
- Combined AG-IG device: A device including both IG and AG functionality, that exposes HNI-INI-AG to OITFs in the residential network.
- OITF STB: A set top box containing an OITF. It connects to the legacy TV using some non-OIPF specified mechanism, such as HDMI or SCART.
- OITF TV: A TV containing an OITF.

Optionally, the OITF STB or the OITF TV may act as a DLNA DMS to make OIPF services available to DLNA devices. They may also act as a DLNA DMP to access content from other DLNA devices on the home network.

5.2.3 AG-IG, OITF-IG, Multiple OITFs

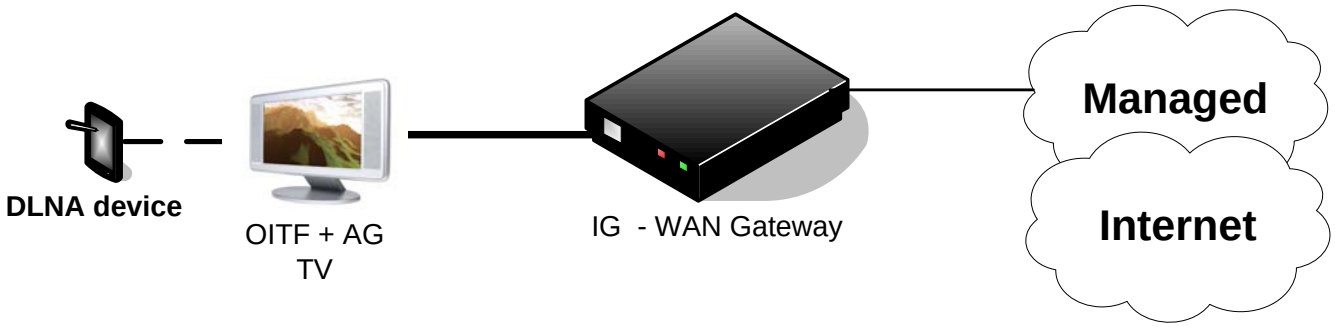


This deployment supports managed and unmanaged services, and DAE and PAE applications. The following devices are deployed:

- A WAN Gateway.
- Combined AG-IG device: A device including both IG and AG functionality, that exposes HNI-INI-AG to OITFs in the residential network.
- OITF STB: A set top box containing an OITF. It connects to the legacy TV using some non-OIF specified mechanism, such as HDMI or SCART.
- IG-OITF STB: A set top box containing both an IG and an OITF.
- OITF TV: A TV containing an OITF.

In this deployment, there are multiple IGs. Only one IG can be active in the residential network at any point in time. The ISIM application must always be in the AG-IG in this case.

5.2.4 Combined OITF-AG TV and IG-WAN Gateway ("headed configuration")



This deployment supports managed and unmanaged services, and DAE and PAE applications. The following devices are deployed:

- Combined IG-WAN Gateway: A single physical device including an IG and WAN Gateway functionality.
- Combined AG-OITF TV: A TV including both an OITF and an AG.

Optionally, the OITF-AG TV MAY act as a DLNA DMS to make OIPF services available to DLNA devices. It MAY also act as a DLNA DMP to access content from other DLNA devices on the home network

5.3 Remote UI Server (informative)

PAE applications MAY offer a CE-HTML remote UI to other devices. The org.oipf.uiserver package allows applications to register as providing a remote UI. When an application registers, it provides information about itself in the XML format defined by the [CEA-2014-A] remote UI. The PAE implementation listens for UPnP search requests for the schema "urn:oipf-org:device:ag:1" as defined in [OIPF_PROT2] section 10.1.1.2. When such a request is received, the implementation combines the remote UI information provided by all currently registered applications and returns it.

6 Protocols

6.1 Broadcast Channel Protocols

The broadcast channel protocols defined by clause 6.2 of [GEM] are NOT included in the present document.

6.2 Interaction Channel Protocols

The following protocols SHALL be supported with the definition in [OIPF_PROT2] superseding that in [GEM].

- TCP
- UDP
- IP
- DNS
- HTTP
- HTTPS

6.3 Transport protocols for application loading over the interaction channel

The [GEM] "File system implemented only via the interaction channel" SHOULD NOT be supported.

Applications loading over the interaction channel SHALL be implemented using signed JAR-files as defined in [JAR].

The FLUTE protocol SHALL be supported as defined in [OIPF_PROT2].

When the application signalling for an application includes one or more FLUTE transport protocol descriptors, (see clause "Extensions to XML AIT"), the following SHALL apply;

- The implementation SHALL attempt to connect to each specified FLUTE session or sessions (unless already connected for another application)
- Requests to load files SHALL be checked against the files available through FLUTE before fetching them from a unicast network
- When a FLUTE session is not used by any running applications, the session SHALL be closed. It is implementation dependent whether this happens immediately or at some later time when implementation resources used by the closed session MAY be needed for other purposes. Closed FLUTE sessions MAY be subsequently re-used however the implementation SHALL ensure the contents are current – as if a new session had been created.

6.4 IPTV Protocols

6.4.1 Streaming Protocols

The following protocols SHALL be supported as defined in [OIPF_PROT2].

- RTP
- IGMP
- RTSP

6.4.2 Metadata Protocols

The following protocol SHALL be supported;

- DVB-STP as required by [OIPF_META2]

6.4.3 Content Download Protocols

The following protocols SHALL be supported as defined in [OIPF_PROT2].

- HTTP

6.5 Home Network Protocols

This is the procedure for AG discovery defined in [OIPF_PROT2].

7 Content formats

7.1 Static formats

Requirements on still image support are defined in referenced specifications. The following is a summary;

- JPEG support is required by [GEM] and [PBP 1.1] with the precise details defined in [GEM]
- PNG support is required by [GEM] and [PBP 1.1] with the precise details defined in [GEM]
- GIF support is required by [PBP 1.1]

The optional MPEG-2 I-frames and video “drips” in [GEM] SHALL NOT be used.

7.2 Streaming formats

The following SHALL be supported for streaming content as defined by [OIPF_MEDIA2].

- H.264/AVC
- HE-AAC

The following SHALL be supported as monomedia formats for audio clips as defined by [OIPF_MEDIA2]

- HE-AAC

MPEG-1 layer 3 MAY be supported for streaming content and for audio-clips as defined by [OIPF_MEDIA2].

Subtitles SHALL be supported as defined by [OIPF_MEDIA2].

OIPF HTTP Adaptive Streaming MAY be supported as defined by [OIPF_HAS2].

7.3 Fonts

7.3.1 Resident Fonts

The RNIB/DTG “Tiresias” font or equivalent SHALL be supported with the “Basic Euro Latin” character set defined in [MHP].

Other resident fonts MAY be supported as required.

7.3.2 Downloadable Fonts

The OpenType format as defined by [GEM] SHALL be supported. The PFR font format as required by [GEM] is NOT required to be supported.

8 Void

9 Application Model

9.1 Introduction (informative)

The model for procedural applications is that defined by [GEM]. Individual applications are grouped into services. Services may include streaming video and audio but this is not required. Services which contain only applications are permitted and indeed may be more common in this context. The organisation that packages a set of applications into a service is responsibility for ensuring that those applications behave reasonably when deployed together. Applications within a service carry a control code, one of which is AUTOSTART. Applications with this control code will be started automatically when the service is selected.

An application gateway becomes aware of services either from the IPTV service discovery process (see [OIPF_ARCH2]) or as a result of an API call made by an application. This signalling supports applications that run all the time (called "unbound applications" in [GEM]), and applications which are bound to a scheduled content service. Practically this signalling does not support applications which are bound to a particular item of content in a scheduled content service. Applications are signalled by information placed in an ApplicationList which may be placed in the following 3 places;

- In the SD&S service provider discovery record for the service provider who owns or subsidises the device including the application gateway (called the "subsidising service provider" in [GEM]). Any such applications, which are signalled as needing to be started automatically, will be started when the SD&S is first processed. Such applications are unbound applications and may run all the time. See also section 3.2.3.1 of [OIPF_META2].

- In a Package entry in SD&S. Such applications are bound to the set of services in the package and run while a service from that package is selected.

- In an IPService entry in SD&S. Such applications are bound to the service (or services) where they are included and only run while a service is selected where those SD&S entry includes them.

Additionally an application offering a user interface to on-demand content may associate one or more applications with individual items of content using an API which associates the application signalling information with the locator for the on-demand content.

There are a number of different API calls which an application can use to add or update the services known to an application gateway.

- An application may install a service (and the applications it contains) in persistent memory in the application gateway (known as "stored services" in [GEM]).

- An application may select a service by specifying the location of a file containing the signalling information for that service, (e.g. using an HTTP URL pointing to a file on a server). Such a service will just run without being installed in persistent memory. (These are known as "Applications loaded from the interaction channel" in [MHP].)

- An application may install services containing unbound applications by passing in the same information as would be acquired from SD&S.

The present document only requires application signalling that is in an XML format, either distributed as part of SD&S or distributed as a stand-alone file. Practically this signalling is not suitable for dynamic uses such as applications which are related to a particular content item in a scheduled content service. In the present document, the in-band MPEG-2 table based signalling from MHP could be used for this purpose however it is optional for both networks and application gateways.

In addition to the mechanisms specified above, applications and services containing them may be included as part of the basic software of the application gateway and start automatically when power is applied. Such applications would be updated by software download. Such applications would likely be unbound applications running all the time. For networks where SD&S is not deployed, one possibility would be for an application to load an initial set of applications by some other means, e.g. from a service provider specific URL, either unicast or multicast.

In the present document, all procedural applications are distributed in JAR files, one JAR file per application. Due to the file format, it is necessary to download the end of the JAR file before any of the individual files inside can be accessed, practically therefore the entire JAR file is downloaded before an application starts. This is in contrast to the broadcast

object carousel based systems where individual class files can be downloaded as required. These JAR files may be distributed by unicast (HTTP) or multicast (FLUTE).

Applications and their associated metadata may be stored in persistent, non-volatile storage in the terminal or downloaded from the network into volatile memory when required. The following 3 possibilities are defined;

- Both application and metadata stored in persistent storage. These applications can run with no further access to the network. These are called Stored Applications in GEM.
- Applications stored in persistent storage whose metadata is not stored. For these applications, the persistent storage effectively functions as a cache. The metadata for the application is retrieved from the network. At this point the implementation determines that the correct version of the application is held locally in persistent storage.
- Neither application nor metadata is stored in persistent storage. In this case once the metadata is downloaded, the corresponding application file(s) must also be downloaded. At least the JAR file must be downloaded before the application can run.

9.2 Broadcast Applications

In the present document, what [GEM] calls “broadcast applications” are applications which are associated with one or more scheduled content services. These SHALL be supported using the “XML Encoding for AIT” found in Annex AR of [MHP] as follows;

- Signalling them in a Package entry in SD&S. Such applications are bound to the set of services in the package and are permitted to run while any service from that package is selected.
- Signalling them in an IPService entry in SD&S. Such applications are bound to the service (or services) where they are included and only run while any service is selected where those SD&S entry includes them.

These SHALL also be supported where an application offering a user interface to on-demand content associates one or more applications with individual items of content manually.

Applications MAY also be associated with scheduled content using the MPEG-2 table based encoding of the AIT defined in [MHP]. Support for this is OPTIONAL in both networks and application gateways.

9.3 DVB-J Model

Application gateways are NOT required to monitor for changes in SD&S frequently enough to change applications at boundaries between individual content items in a scheduled content service. Effectively dynamic update of a service's application signalling is not supported. The language in [GEM], section 9.2 about the packaged media target also applies to application gateways.

Note: SD&S information changes only if a new service is added. This is not happening too frequently and a check once a day is considered to be sufficient.

9.4 Stored and Cached Applications

Stored applications SHALL be supported.

Cached applications MAY be supported.

9.5 Unbound Applications

Unbound applications SHALL be supported.

10 Application Signalling / Metadata

10.1 XML AIT

The application description SHALL be the XML encoding of the AIT as defined in [MHP] annex AR "XML encoding for AIT" and modified by [OIPF_META2] with the following modifications;

Table 1: Status of XML AIT Descriptors and Elements

Descriptor or Element in MHP	Summary	Status or comments in present document
AR.3.1 ApplicationList	List of applications	Required
AR.3.2 Application	Name, identifier, type specific descriptor, provider descriptors	Required
AR.3.3 ApplicationIdentifier	2 numbers	Required
AR.3.4 ExternalApplicationIdentifier	Already running applications not signalled in this service	This SHOULD NOT be used.
AR.3.5 ApplicationDescriptor	Numerous application attributes	Required
AR.3.6 VisibilityDescriptor	Attribute – indicate if application can be visible to users and/or other applications	Required
AR.3.7 IconDescriptor	Icon for application	The filename in the IconDescriptor SHALL either be a relative filename within the jar file carrying the application or SHALL be an HTTP URL.
AR.3.8 AspectRatio	Preferred aspect ratio for applications	Required
AR.3.9 MhpVersion	Specification version	As defined by GEM for the application version fields in clause 10.4.3 "Content of the application description"
AR.3.10 StorageCapabilities	Can the application be stored or cached	Support for stored applications required.
AR.3.11 StorageType	Enumeration used in AR.3.10	As AR.3.10
AR.3.12 ApplicationType	Application type	The "DVB-J" application type SHALL be used for GEM-IPTV applications. The "OIPTV-headless" type SHALL be used for headless applications.
AR.3.13 DvbApplicationType	Enumeration for AR.3.12	Application type DVB-J SHALL be used for GEM-IPTV applications.
AR.3.14 ApplicationControlCode	Enumeration for AR.3.5	Required except for REMOTE (see external application identifier)

Descriptor or Element in MHP	Summary	Status or comments in present document
AR.3.15 ApplicationSpecificDescriptor	Container	Required
AR.3.16 DVBJDescriptor	Application location	The first location element SHALL point to a JAR file containing the first class of the application. No other JAR files SHALL be referenced.
AR.3.17 ApplicationStructure	Classpath and initial class for use with AR.3.16	This SHALL NOT be used. (The information is found in the jar file manifest).
AR.3.18 AbstractIPService	Supports grouping of unbound applications	Required
AR.3.19 UnboundApplicationDescriptor	Unbound application support	Required

Where FLUTE is supported as defined in [OIPF_PROT2], the FLUTESessionDescriptor defined in annex B.6 of [OIPF_META2] SHALL be supported. Where FLUTE is not supported, this descriptor SHALL be silently ignored.

10.2 Stored and cached applications

For stored and cached applications, the application description file SHALL be ignored. The JAR file or its entire contents SHALL be stored.

11 The Java Platform

11.1 Fundamentals

The PAE Java Platform is based on the IPTV target of the DVB-GEM specification [GEM] which is based on the Personal Basis Profile 1.1 [PBP 1.1] and Java TV 1.1 [JAVA TV] as described below.

In the case of a conflict, the following precedence rules apply:

1) The normative guarantees of “Personal Basis Profile 1.1” SHALL always take precedence.
2) The normative guarantees of “Java TV 1.1.” SHALL always take precedence, except when in conflict with rule 1.
3) The normative guarantees of GEM-IPTV take precedence except when in conflict with rules 1 and 2 above.

The PAE Java Platform is based on the “Personal Basis Profile 1.1”. PAE implementations SHALL fully comply with the “Personal Basis Profile 1.1”.

All PAE Implementations SHALL be fully compliant with:

- [PBP 1.1] “Personal Basis Profile 1.1”
- [CDC 1.1] “Connected Device Configuration 1.1”
- [FP 1.1] “Foundation Profile 1.1”

The PAE Java Platform is based on “Java TV 1.1”. PAE implementations SHALL fully comply with “Java TV 1.1“.

- [JAVA TV] “Java TV 1.1.1”

  NOTE: Failure modes of JavaTV in devices without a display are summarized in Headless behaviour of UI-related APIs (informative).

The namespaces of the specifications above SHALL be protected, it is NOT permitted to implement any additional method, which is not defined in these specifications. Subsetting of any class or package is also NOT permitted.

GEM 1.3

The PAE Java Platform is based on IPTV target of the DVB-GEM 1.3 specification [GEM] and the “Application Gateway and Media Server Fragment” specification [HEADLESS].

PAE implementations in devices with a display SHALL be fully compliant with the mandatory requirements of the IPTV target of the DVB-GEM 1.3 specification with the clarifications, definitions and extensions contained in this document.

When 3D is supported, PAE implementationsin devices with a display SHALL also be fully compliant with the mandatory requirements of the Plano-stereoscopic 3DTV GEM Profile [GEMS3D]

PAE implementations in devices without a display SHALL be fully compliant with the requirements of the “Application Gateway and Media Server Fragment” specification.

PAE applications SHALL NOT define classes in the namespace of any package specified by the PAE specification. The fully qualified class name of any class defined by an application SHALL not start with `“java.”, “javax.microedition.”, javax.crypto.”, “javax.net.”, “javax.security.”, “javax.media.”, “javax.tv.”, “org.davic.”, “org.dvb.”, or “org.havi.” or “org.oipf.”`.

11.2 Extensions and mappings to GEM APIs

11.2.1 Broadcast Transport Protocol Access API (org.dvb.dsmcc)

Instances of DSMCCObject corresponding to files in a JAR file SHALL be supported as defined by clause P.3 of [GEM].

Instances of ServiceDomain SHALL correspond to JAR files. Use of the attach(Locator) method with a Locator referencing a JAR file delivered using HTTP SHALL be supported.

The present document does not define support for instances of DSMCCStream and DSMCCStreamEvent corresponding to files in a JAR file.

11.2.2 Application Listing and Launching API (org.dvb.application)

This API SHALL be mapped onto the XML encoding of the AIT.

11.2.3 Streaming Media APIs

If the Locator, MediaLocator or URL passed to the methods listed below references a content access streaming descriptor (as defined in annex E.2 of [OIPF_DAE2]) which contains one or more content items then the client SHALL use the first such content item as the source of the media to be presented by each of the APIs listed. This applies to the following methods;

- javax.tv.service.SIManager.getService(...)
- javax.media.Manager.createDataSource(...)
- javax.media.Manager.createPlayer(...)

11.2.4 GEM 3D API

The org.dvb.stereoscopicpackages SHALL be supported as defined in [GEMS3D].

11.3 APIs defined by this Volume

11.3.1 Content and Service Protection API

This is the org.oipf.drm package defined in Annex G.

When the permission request file requests the permission to communicate with a CA system for any CA system ID and this is granted, a DRMPermission SHALL be created with name "*".

11.3.2 User Authentication API

This is the org.oipf.auth package defined in Annex E.

When an application is started, it SHALL be given an instance of org.oipf.auth.UserAuthenticationPermission where the domain is the domain from which the application was delivered.

11.3.3 UI Server API

This is the org.oipf.uiserver package defined in Annex F.

11.3.4 Content download API

This is the org.oipf.download package defined in Annex C.

If the locator references a content access download descriptor (as defined in annex E.1 of [OIPF_DAE2]) which contains one or more content items then the client SHALL use the data inside the abstract content access descriptor to initiate the download of those content items. Where multiple content items are to be downloaded, all items SHALL be downloaded in an order as defined by the client.

11.3.5 Service API

This is the org.oipf.service package defined in Annex D.

11.4 PVR APIs

NOTE: The basic APIs for PVR are included as a result of the reference to [DVR] in clause 4316 – "PVR".

This is the following packages;

- the org.dvb.tvanytime.pvr package and sub-packages from [MHP-PVR]
- the org.oipf.pvr package defined in Annex H

11.5 Content referencing

The following table lists the types of locators defined in clause 14 (“system integration aspects”) and their mapping to the APIs required by [GEM].

Table 2: Mapping of GEM Clauses Relating to Content Referencing

GEM clause	Mapping
11.11.1 Transport stream	Not required for GEM-IPTV target
11.11.2 Network	Not required for GEM-IPTV target
11.11.3 Bouquet	Not included in GEM
11.11.4 Service	As specified by GEM.
11.11.5 DVB event	Not required for GEM-IPTV target
11.11.6 MPEG elementary stream	As specified by GEM.
11.11.7 File	Implementations SHALL support the use of URLs to reference files in JAR files. The details of this MAY be implementation specific.
11.11.8 Directory	Implementations SHALL support the use of URLs to reference directories in JAR files. The details of this MAY be implementation specific.
11.11.9 Drip feed decoder	Not required for GEM-IPTV target
11.11.10 Irrelevant	Not relevant.
11.11.11 Methods working on many locator types	As specified by GEM.
11.11.12 Support for the HTTP Protocol in DVB-J	As specified by GEM.
11.11.13 MHP Applications	As specified by GEM.

12 Security

12.1 Authentication of Applications

Application authentication SHALL be done using the signing mechanisms defined for JAR files in [JAR]. The MHP application authentication mechanism SHOULD NOT be supported by the current specification.

12.2 Permission request file

NOTE: The semantics of the “capermission” element are modified as defined by clause1 11.3.5 Service API.

12.3 Security Policy for Applications

The permission request file defined in [GEM] SHALL be supported except as follows;

- The credentials mechanism SHOULD NOT be supported
- Permissions where the corresponding permission class is not included in the present document SHALL be parsed but then ignored.

12.4 Certificate Management

The certificate revocation mechanism defined in [MHP] SHOULD NOT be supported. Certificate revocation MAY optionally be checked using [OCSP].

The root certificate management mechanism defined in [MHP] SHOULD NOT be supported. No replacement is defined by the present document. Updating of root certificates MAY be performed by software update.

13 Graphics reference model

The following apply in addition to the requirements defined in [GEM].

Graphics resolutions for standard definition (as required by [GEM] clause G.1.1) SHALL be as defined in [OIPF_DAE2].

14 System integration aspects

The following table lists the types of entity that it SHALL be possible to address by locators in the present document and any corresponding text representation.

Table 3: Locators and Corresponding Text Representations

Entity	Text Representation	Comment
Service	For services delivered via unicast, "rtsp:" URL defined in [BCG]. For services delivered via multicast, “rtp:” and “udp” URLs defined in [BCG].	Relevant for devices with a display and for devices without a display but with media handling capabilities.
Service Domain	“http:” or “https:” URL	Used to reference a JAR file
MPEG elementary stream	No standardized text representation	Relevant for devices with a display and for devices without a display but with media handling capabilities.
File	"file:", "http:" and "https:" URLs as referred to in [GEM].	
Directory	"file:", "http:" and "https:" URLs as referred to in [GEM].	

NOTE: FLUTE carousels do not appear as a file system. Hence they do not have a unique URL.

NOTE: “rtp:”, “udp:” and “rtsp:” are considered as transport dependent locators. The present document does not define support for transport independent locators.

15 Detailed Profile Definitions

The following table lists which clauses of the present document are REQUIRED or OPTIONAL for the 3 different device types addressed by this volume.

Table 4: Platform Profile Definitions in this Volume

Area	Clause	Devices with Display	Devices without Display	
			Without media handling	With media handling
	4.1.1 DVB-GEM Compliance	M	-	-
Protocols	6.1 Broadcast Channel Protocols	-	-	-
	6.2 Interaction Channel Protocols	M	M	M
	6.3 Transport protocols for application loading over the interaction channel	M	M	M
	6.4.1 Streaming Protocols	M	-	M
	6.4.2 Metadata Protocols	M	M	M
	6.4.3 Content Download Protocols	O	-	O
	6.5 Home Network Protocols	O	M	M
Content formats	7.1 Static formats	M	-	-
	7.2 Streaming formats	M	-	O
	7.3.1 Resident Fonts	M	-	-
	7.3.2 Downloadable Fonts	M	-	-
Application Model	9.2 Broadcast Applications	M	-	-
	9.3 DVB-J Model	M	M	M
	9.4 Stored and Cached Applications	M	M	M
	9.5 Unbound Applications	M	M	M
Application Signalling	10.1 XML AIT	M	M	M
	10.2 Stored and cached applications	M	M	M
Java Platform	11.1 Fundamentals	M	M	M
	11.2 Extensions and mappings to GEM APIs	M	M	M
	11.3.1 Content and Service Protection API	M	-	M
	11.3.2 User Authentication API	M	M	M
	11.3.3 UI Server API	M	M	M
	11.3.4 Content download API	O	-	O
	11.3.5 Service API	M	-	M

Area	Clause	Devices with Display	Devices without Display	
			Without media handling	With media handling
	11.4 PVR APIs	O	-	O
Security	12.1 Authentication of Applications	M	M	M
	12.2 Permission request file	M	M	M
	12.4 Certificate Management	M	M	M
	13 Graphics reference model	M	M	M
	14 System integration aspects	M	M	M
	16 PVR	O	-	O
	17 Minimum Terminal Capabilities			

Key	
-	Not applicable/Not required/ Clause does not contain normative requirements
O	Optional feature
M	REQUIRED feature in the GEM terminal

Table 5: Applicability of GEM Specification Sections

Area	GEM Specification Section	GEM IPTV Target	PAE Devices with display	PAE Devices without Display	
				Without media handling	With media handling
Static Formats					
Bitmap pictures	7.1.1.3, "PNG" and 15.1, "PNG – restrictions"	M	M	-	-
	7.1.1.3, "PNG" without restrictions	-	-	-	-
	7.1.1.4, "GIF"	-	M Required by PBP	-	-
	7.1.2, "MPEG-2 I-Frames"	O	SN	-	-
	7.1.1.2, "JPEG" + 15.3, "JPEG – restrictions"	-	-	-	-
	7.1.1.2, "JPEG" without restrictions	M	M	-	-

Area	GEM Specification Section	GEM IPTV Target	PAE Devices with display	PAE Devices without Display	
				Without media handling	With media handling
Audio clips	7.1.4, "Monomedia format for audio clips"	M	M-FE	-	O-FE
Video drips	7.1.3, "MPEG-2 Video "drips"	O	SN	-	-
Text encoding	7.1.5, "Monomedia format for text"	M	M	M	M
Media Streaming formats					
Video	7.2.2, "Video"	M	M-FE	-	-
Audio	7.2.1, "Audio"	M	M-FE	-	O-FE
Subtitles	7.2.3, "Subtitles"	-	O	-	-
Fonts					
Built in	Character, set see annex E Metrics see annex D Face: UK RNIB "Tiresias"	O	M	-	-
Downloadable	7.4.1, “PFR”	M	SN	-	-
	7.4.2, “OpenType”	O	M	-	-
Broadcast channel protocols					
	6.2.2, "MPEG-2 sections"	O	Not visible to GEM applications	-	Not visible to GEM applications
	6.2.5, "Object carousel"	O	M/- Partial functional equivalent	M/- Partial functional equivalent	M/- Partial functional equivalent
	IP Multicast stack based on: 6.2.6, "Protocol for delivery of IP multicast over the broadcast channel", 6.2.7, "Internet Protocol (IP)", 6.2.8, "User Datagram Protocol (UDP)", 6.2.10, “IP signalling”	-	-	-	-
Interaction channel protocols					

Area	GEM Specification Section	GEM IPTV Target	PAE Devices with display	PAE Devices without Display	
				Without media handling	With media handling
TCP/IP	6.3.3, "Transmission Control Protocol", 6.3.2, "Internet Protocol"	M	M	M	M
UDP/IP	6.3.2, "Internet Protocol", 6.3.9, "User Datagram Protocol"	M	M	M	M
HTTP	6.3.7.1, "HTTP 1.1"	O	M	M	M
	6.3.7.2, “MHP profile of HTTP 1.0”	M	-	-	-
DSMCC-UU RPC	6.3.4, "UNO-RPC", 6.3.5, "UNO-CDR", 6.3.6, "DSM-CC User to User"	-	-	-	-
DNS	6.3.10, "DNS"	M	M	M	M
HTTPS	6.3.7.3, “HTTPS”	M	M	M	M
Interaction Channel File System	6.4.1, “File system implemented only by the interaction channel”	O	-	-	-
DSMCC / HTTP hybrid	6.4.2, “Hybrid between broadcast stream and interaction channel”	O	-	-	-
IPTV	5, “Basic architecture”	M	-	-	-
Application Model					
Application Model	All parts of clause 9, “Application model” except those clauses (and their subclauses) identified below	M	M	M	M
	9.3 ”DVB-HTML model”	O	SN	SN	SN
	9.7, “Lifecycle of internet access applications”	O	M/O Only web browser support is REQUIRED	-	-
	9.9, “Stored and Cached applications”	O	M/O Stored application support REQUIRED	M/O Stored application support REQUIRED	M/O Stored application support REQUIRED
	9.13, “Unbound Applications”	-	M	M	M

Area	GEM Specification Section	GEM IPTV Target	PAE Devices with display	PAE Devices without Display	
				Without media handling	With media handling
Application Signalling					
Application Signalling	10, “Application signalling”	M	M-FE XML encoding of AIT does not support full syntax of binary encoding.	M-FE XML encoding of AIT does not support full syntax of binary encoding.	M-FE XML encoding of AIT does not support full syntax of binary encoding.
DVB-J					
	All parts of clause 11, “DVB-J platform” except those clauses (and their subclauses) identified below	M	M	Subset as defined by [HEADLESS]	Subset as defined by [HEADLESS]
	11.4.1 “ HAVi UI Widgets”	O	O	-	-
	11.5.2, "Support for Multicast IP over the Broadcast Channel"	-	-	-	-
	11.5.3, "Support for IP over the Return Channel"	M	M	M	M
	11.5.4, "MPEG-2 Section Filter API"	O	SN	SN	SN
	11.5.5, "Mid-Level Communications API" as modified by 11.5, "Data access APIs"	M	M	M	M
	11.5.7 “File storage device access”	O	O	O	O
	11.6.3, "Tuning API"	O	-	-	-
	11.6.4, "Conditional access API"	-	-	-	-
	11.6.6, ”Service discovery and selection for IPTV”	O	M	M	M
	11.6.7, “Integration between protocol independent SI API and TV-Anytime”	O	M/- Mandatory if broadband content guide is supported otherwise not applicable	M/- Mandatory if broadband content guide is supported otherwise not applicable	M/- Mandatory if broadband content guide is supported otherwise not applicable

Area	GEM Specification Section	GEM IPTV Target	PAE Devices with display	PAE Devices without Display	
				Without media handling	With media handling
	11.7.4, "Basic MPEG concepts"	O	-	-	-
	11.7.6 “Content referencing”	O	M	M	M
	11.7.7, "Common error reporting"	O	-	-	-
	11.7.8, “Plug-in APIs”	O	O	O	O
	11.7.9, “Provider API”	O	O	O	O
	11.7.10, “Content referencing for IPTV”	O	M	-	M
	11.7.11, “TV-Anytime content referencing and metadata	O	M/- Mandatory if broadband content guide is supported otherwise not applicable	M/- Mandatory if broadband content guide is supported otherwise not applicable	M/- Mandatory if broadband content guide is supported otherwise not applicable
	11.8.2, "APIs for return channel security"	M	M	M	M
	11.8.3, "Additional permissions classes"	O	-	-	-
	11.8.6, “DVB Extensions for Cryptography”	O	O	O	O
	11.9.5.2, “JDOM”	O	M	M	M
	11.9.6, “MHP terminal hardware API”	O	O	O	O
	11.11.1 “Transport stream”	O	-	-	-
	11.11.2 “Network”	O	-	-	-
	11.11.4.3, “Content referencing for IPTV”	M	M	-	M
	11.11.5 ”DVB event”	O	-	-	-
	11.11.9 “Drip feed decoder”	O	SN	SN	SN
	11.12.2, “Stored services”	O	M	M	M

Area	GEM Specification Section	GEM IPTV Target	PAE Devices with display	PAE Devices without Display	
				Without media handling	With media handling
	11.15, “APIs defined in OCAP”	O	M/O Only org.ocap.service is REQUIRED	M/O Only org.ocap.service is REQUIRED	M/O Only org.ocap.service is REQUIRED
	11.14, “Internet Access”	O	M/O Only web browser support is REQUIRED	-	-
	17, “Internet access clients”	O	M/O Only web browser support is REQUIRED	-	-

Key	
-	Not applicable/Not required
O	Optional feature
SN	Feature SHOULD not be supported
M	Mandatory feature
M-FE	Mandatory feature – functional equivalent
O-FE	Optional feature – functional equivalent

The following table gives an overview of the functional equivalents that are available in the GEM-IPTV target and are defined in the PAE.

Table 6: Summary of Functional Equivalents (Informative)

Name	GEM Clause(s)	MHP Definition	Details
Arch	5, "Basic architecture"	MHP [1], clause 5, "Basic Architecture"	See section 5.1 Architecture
Carousel	6.2.5, "Object carousel"	MHP [1], clause 6.2.5, "DSMCC User-to-user Object Carousel" as modified by clause 15.6.1.1, "Carousel"	No complete functional equivalent defined. JAR files are a partial functional equivalent. See section 6.3 Transport protocols for application loading over the interaction channel
IP MPE	6.2.6, "Protocol for delivery of IP multicast over the broadcast channel"	MHP [1], clause 6.2.6, "DVB Multiprotocol Encapsulation"	Not applicable

Name	GEM Clause(s)	MHP Definition	Details
SI	6.2.9, "Service information"	MHP [1], clause 6.2.9, "DVB Service Information" DVB Service Information as defined in EN 300 468 [4] and ETR 211 [11].	DVB-SD&S and BCG.
	11.6.1, "Signalling-bound service information API"	MHP [1], clause 11.6.1, "DVB Service Information API" MHP [1], clause 11.11.3, "Bouquet"	Not defined.
	Annex O, "Integration of the JavaTV SI API"	MHP [1], annex O, "Integration of the JavaTV SI API"	MHP Annex AP mapping between JavaTV SI API and SD&S / BCG.
Broadcast IP signalling	6.2.10, "IP signalling"	MHP [1], clause 6.2.10, "IP signalling"	Not applicable
IPTV Protocols	6.5, “IPTV protocols”	MHP [1], clause 6,5, “IPTV protocols”	RTP, RTSP, IGMP, DVB-STP as defined in section 6.4 IPTV Protocols.
Audio	7.2.1, "Audio" At least 1 audio format.	MHP [1], clause 7.2.1, "Audio"	HE-AAC - see section 7.2 Streaming formats
Video	7.2.2, "Video" At least 1 video format.	MHP [1], clause 7.2.2, "Video"	H.264/AVC - see section 7.2 Streaming formats
Subtitles	7.2.3, "Subtitles"	MHP [1], clause 7.2.3, "Subtitles"	Optional in the present document
	11.4, "Presentation APIs" - classes related to subtitles	MHP [1], clause 11.4.2.5.1, classes related to subtitles	If no subtitle format selected then APIs do nothing.
Audio Clips	7.1.4, "Monomedia format for audio clips"	MHP [1], clause 7.1.4: Functional equivalent to MPEG-1 Audio (Layer 1 and 2) ISO/IEC 11172-3	HE-AAC REQUIRED and MPEG-1 layer 3 OPTIONAL - see section 7.2 Streaming formats.
Resident Fonts	7.3, "Resident fonts"	MHP [1], clause 7.3 At least the font Tiresias [80] shall be provided.	Tiresias or equivalent with extended latin character set.
Downloadable Fonts	7.4, "Downloadable fonts"	MHP [1], clause 7.4 PFR0 as in DAVIC 1.4.1p9 [3] OpenType® as defined in ISO/IEC 14496-18 [115] as the font format for MPEG-4	OpenType REQUIRED with clarifications from GEM.

Name	GEM Clause(s)	MHP Definition	Details
Application Signalling	10.2, "Program specific information" 10.4, "Application description" 10.5, "DVB-J specific application description"	MHP [1], clause 10	XML encoding of AIT defined in MHP Annex AR.
	11.7.2, "Application discovery and launching APIs"	MHP [1], clause 11.7.2, "Application discovery and launching APIs"	Same mapping as MHP. See clause 11.2.2 - Application Listing and Launching API (org.dvb.application).
Application Authentication	12.2, "Authentication of applications" NOTE 1: See also text in clause 12.6.2.6, "Credentials" and text in clause 12.9, "Certificate management" NOTE 2: See also text in clause 12.9.2, "Root certificate management."	MHP [1], clause 12.2, "Authentication of applications"	Signed Jar files.
	12.9.1, "Certificate Revocation Lists"	MHP [1] clause 12.9.1, "Certificate Revocation Lists"	Optional in the present document.
Conditional Access	11.4, "Presentation APIs," classes related to conditional access	MHP [1], clause 11.4.2.5.1, classes related to conditional access	Failure modes due to content protection are included.
	11.6.4, "Conditional access API"	MHP [1], clause 11.6.4, "Conditional Access API" MHP [1], clause 14.10, "CA system"	Not included in the present document.
Content Referencing	11.7.6, "Content referencing"	MHP [1], clause 11.7.6, "Content Referencing" This API is formed of DAVIC 1.4.1p9 [3] Locator, DvbLocator, javax.tv.locator, org.dvb.locator.packages	Supported as defined in GEM. Classes corresponding to “dvb:” URI scheme not included.
	11.11.11, "Methods working on many locator types"	MHP [1], clause 11.11.11, "Methods working on many Locator types"	There are no transport independent locators defined in the present document.
	14.1, "Namespace mapping"	MHP [1], clause 14.1, "Namespace mapping (DVB Locator)"	See section 14 System integration aspects.

Name	GEM Clause(s)	MHP Definition	Details
	14.9 Content referencing for IPTV 14.10, "Service identification"	MHP [1], clause 14.9, "Service identification"	Not applicable.
Graphics Resolution	D.2, "Horizontal resolution"	MHP [1], clause D.3.4.2, "Horizontal resolution"	
	G.1.1, "Device resolution for Standard Definition"	MHP [1], clause G.1.1, "Device capabilities"	See [OIPF_DAE2].
	G.4, "Resident fonts and text rendering"	MHP [1], clause G.4, "Resident fonts and text rendering"	
Text Wrapping	D.3, "Text wrapping setting is true"	MHP [1], clause D.3.7.2, "Text wrapping setting is true"	
RCMM	12.9.2, "Root certificate management"	MHP [1], clause 12.9.2, "Root Certificate Management"	Not defined in the present document.
Active Format Descriptor	N.1, "Active Format Definition"	Described in the present document, clause N.1.1, "MHP Signalling for Active Format Definition"	Not defined in the present document.

Key	
M	Functional equivalent required
M*	MHP definition of the functional equivalent required
O	Optional feature, no functional equivalent required
-	Not required/Not applicable

References	
[1]	ETSI TS 102 727 v1.1.1, “Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.2.2”
[3]	DAVIC 1.4.1 Specification part 9, “Information Representation”
[4]	ETSI EN 300 468, “Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB system”
[11]	ETSI TS 101 211 v1.9.1, “Digital Video Broadcasting (DVB); Guidelines on implementation and usage of Service Information (SI)”
[80]	Tiresias: "The RNIB/DTG "Tiresias" font version 8.03".
[115]	ISO/IEC 14496-18, “Information Technology – Coding of audio-visual objects – Part 18: Font compression and streaming”

16 PVR

Where local PVR functionality is exposed to the PAE, this SHALL be done as specified by [DVR]. The details which GEM recording specifications are required to provide by that specification are defined in the following tables.

16.1 Mandatory Responsibilities

Table 7: Responsibilities of GEM Recording Specifications

Responsibility	Definition
Which types of stream are to be considered as “recordable streams”.	All streams defined by [OIPF_MEDIA2] SHALL be considered recordable. To the extent that the terminal supports subtitles, subtitle streams SHALL be considered recordable.
Minimum capabilities for the number of steams (or number of streams of each type) that a GEM recording terminal must be able to record.	For scheduled recording, the simultaneous recording of at least one video elementary stream, at least two audio elementary streams and at least two subtitle streams to the extent they are supported.
The definition of which applications are recordable in both scheduled and timeshift recording (which need not be the same).	Not applicable. The present document does not require support for applications in-band in broadcast services.
The requirements on a GEM recording terminal to monitor for dynamic data and behaviour of applications during scheduled and timeshift recording (which need not be the same).	Not applicable. The present document does not require support for data in-band in broadcast services.
Requirements on reconstructing the dynamic behaviour of recorded applications during playback of scheduled and timeshift recordings (which need not be the same).	Not applicable. The present document does not require support for applications in-band in broadcast services.
The definitions of which streams are to be recorded in scheduled and timeshift recording	If the number of streams of each type present is less than or equal to the limits in the recording capability of the device then all the streams of that type SHALL be recorded. Where more streams of any one type exist than the terminal can record, the decision on what to record SHALL be according to clause 11.4.2.3 of [MHP].
How accurately the expiration period should be enforced by implementations.	There is no requirement for this to be accurately enforced, either by deleting the recording or by making it inaccessible through the API.
The definition of at least one protocol for transmitted time lines	Not defined in the present document.
The conditions when a JMF player or service context has a time-shift buffer attached.	The implementation SHALL support at least one time-shift buffer. If only one video stream is being presented, this time-shift buffer SHALL be associated with that video stream. If more than one video stream is being presented, the present document does not define which stream the time-shift buffer is associated with. The present document does not define mechanisms to associate timeshift buffers with other service contexts.
A mechanism to associate security attributes with individual recording requests	See org.oipf.pvr.RecordingAccessPermissions
The mechanism for resolving parent recording requests including setting the initial state of a parent recording request.	When a request is made to record a group CRID, the terminal SHALL resolve this into its constituent elements.
The events generated during playback when the start and end of a recording a reached.	See Table 8 below.

Table 8: Events During Normal Playback and Resulting Behaviour

Event	Behaviour	Result on screen	Java Event
Fast forward to end of stream when recording is in progress	End of media event generated to any registered applications	Playback continues at rate 1.0 at the end of the stream	EndOfContentEvent
Rewind to beginning of stream	Switch to pause mode	First frame frozen	org.ocap.shared.media.BeginningOfContentEvent
Fast forward to end of stream when recording is not in progress and play to end of stream	End of media generated to any registered applications	Last frame frozen	EndOfMediaEvent

16.2 Optional Responsibilities

The following table identifies where in the present document the optional responsibilities listed in [DVR] can be found or if they are not defined.

Table 9: Optional Responsibilities of GEM Recording Specifications

Responsibility	Definition
Mechanisms for controlling the extent to which one application can read or modify scheduled recordings and completed recordings made by another application.	The requestedRecording and completedRecording parameters in the RecordingProperties.
Sub-classes of RecordingListFilter to filter the list of recordings in ways not supported by the present document.	See org.dvb.tvanytime.pvr.navigation.
Rules governing which recordings an application can access.	As defined in clause Visibility of Recording Requests and Recordings between Applications.
Additional JMF controls to be supported for RecordedServices or the contents of a timeshift buffer. Different sets of JMF controls may be specified for these two cases.	None defined in the present document.
Delays in re-starting applications after the return to normal play if this is believed to improve the end-user experience, for example when repeated cycles of fast-forward / play / fast-forward / play.	When playback leaves trick-mode and returns to normal, terminals MAY delay re-starting applications for up to one minute. The behaviour for this minute is implementation dependent.
a mechanism to permit highly trusted applications to obtain instances of RecordingPermission whose action parameter is “*”	No such mechanism defined in the present document.
that the optional behaviour defined in the class description of ServiceContextRecordingSpec, where the contents of the time-shift buffer are stored when the startTime parameter is in the past, becomes mandatory in that particular GEM recording specification.	Not mandatory in the present document
Mechanisms for automatically removing requests from the list of recordings in a pending state if it appears the recording will never happen.	Once the validity period of a request has expired, a terminal MAY discard recording requests which are still in a pending state.
Mechanisms for automatically removing requests from the list of recordings in a failed state based on some criteria they define.	Once the validity period of a request has expired, a terminal MAY discard failed recording requests.
Any requirements to re-resolve ParentRecordingRequests after the request has first been made and to update the state accordingly	For incompletely resolved CRIDs, the terminal SHALL monitor when an additional resolution information becomes available and SHALL act on that additional information.

16.3 Visibility of Recording Requests and Recordings between Applications and Service Providers

The [DVR] specification requires users of that specification to define a mechanism to manage visibility of recording requests between applications from different organisations – referred to as "RecordingRequest specific security attributes". The mechanism defined in this specification is the following;

- When a recording request is made, the application making the request has to specify which other applications can access that recording request and any resulting recordings as part of the RecordingRequest. Each of these is specified using an instance of the class RecordingAccessPermissions.
- The "RecordingRequest specific security attributes" controlling visibility of a RecordingRequest instance are defined by the "requestedRecording" parameter regardless of the state of the recording request.
- The "RecordingRequest specific security attributes" controlling visibility of a RecordedService are defined by the "completedRecording" parameter.
- How access to a recording request or recorded service is controlled by an instance of RecordingAccessPermissions is defined in the specification for that class.

17 Minimum Terminal Capabilities

PAE devices with a display SHALL comply with the union of the minimum terminal capabilities of [OIPF_DAE2] and [GEM].

18 HTTP Adaptive Streaming

When the AG supports HTTP Adaptive Streaming (HAS), the Player object SHALL accept URL, MediaLocator and DataSource objects pointing to a Media Presentation Descriptor (MPD). The MPD SHALL be delivered with the MIME type as specified for the MPD in [TS26234], i.e. “video/vnd.3gpp.mpd” and in the HAS specification [OIPF_HAS2].

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

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

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

18.1 HAS support

HAS support SHALL be signaled by setting the following System Property:

- **org.oipf.supportsHTTPAdaptiveStreaming**:

The PAE supports HAS (Yes / Null).

Appendix A. Headless behaviour of UI-related APIs (informative)

The behaviour of certain APIs in headless mode (device without a directly connected display) requires a closer consideration. Some APIs were specifically designed for headless operation such as the PBP 1.1 specification, which defines a headless mode and provides methods to enquire the presence of a display.

A.1 PBP

The Personal Basis Profile specification /ref PBP/ introduced a headless operation mode starting in version 1.1.

The class **`java.awt.GraphicsEnvironment`** contains a method **`public static boolean isHeadless()`** which tests whether or not a display *and some form of input device* can be supported in this environment. If this method returns true, a HeadlessException is thrown from areas of the Toolkit and GraphicsEnvironment that are dependent on a display *or input device*.

A.2 JavaTV

The following section discusses the behaviour of JavaTV in headless devices. The analysis shows, that there are well defined modes of operation for JavaTV on a headless device. All methods that handle display related parameters have a defined behaviour for devices without a display. The following section calls out all these methods and gives a short analysis of the behaviour.

`javax.tv.graphics.AlphaColor`

Summary: this just extends java.awt.Color which in PBP includes alpha.

Analysis: no additional functionality above PBP

Conclusion: No special behaviour or spec change needed for headless JavaTV.

`javax.tv.graphics.TVContainer`

Summary: gets the top level container for an Xlet

Analysis: Cannot fail to return a Container unlike javax.microedition.xlet.XletContext.getContainer.

"If the Xlet is the only Xlet that is currently active to invoke this method, it will return an instance of java.awt.Container that is initially invisible, with an undefined size and position."

Spec presumably would allow this to return a "new java.awt.Container" which cannot be used.

XletContext.getContainer says "Calling c.setVisible(true) will make the container visible." This language is missing in TVContainer.

Conclusion: On a headless device, XletContext.getContainer will return an invisible container where the setVisible method fails silently (this is the normal failure mode for this method). No spec change needed for headless JavaTV.

`javax.tv.media.AWTVideoSizeControl` and `MediaSelectControl`

Summary: Controls video scaling and media component selection

Analysis: On a headless device, a JMF player simply fails to support these controls. Player.getControl would return null. Player.getControls would not include these controls in the returned array. I cannot see any language in the JavaTV spec requiring players to support these controls.

Conclusion: On a headless device, these controls are simply not returned by getControl/getControls. No spec change needed for headless JavaTV.

`javax.tv.service.Selection`

Summary: Presentation of (TV) services

Analysis: Selecting a service which a device cannot present will fail with a SelectionFailedEvent. Two reason codes are possible - MISSING_HANDLER and INSUFFICIENT_RESOURCES. The JavaTV specification does not seem to require support for decoding or presenting TV services.

Conclusion: On a headless device, there is no TV service for which

ServiceContext.select(..) would succeed. No spec change needed for headless JavaTV.

`javax.media`

Summary: Presentation of media

Analysis: The entry point to JMF is via Manager.createPlayer and Manager.createDataSource. These throw NoPlayerException and noDataSourceException if no player or datasource can be found.

On a headless device, these exceptions would always be thrown.

(NOTE: Additionally javax.media.Player has a state model. The most relevant states for headless devices are Unrealized and Realizing.

"While Realizing, the Controller performs the communication necessary to locate all of the resources it needs to function". On a headless device, all attempts to leave the Unrealized state would fail. The player would transiently enter the Realizing state and immediately fail with a ResourceUnavailableEvent being sent to any ControllerListeners.)

Conclusion: On a headless device, there is no service for which

Manager.createPlayer(..) would succeed. No spec change needed for headless JavaTV.

Appendix B. Void

Appendix C. Package org.oipf.download

Interface ApplicationDownloadRequest

```
package org.oipf.download;
import java.io.RandomAccessFile;
/**
 * Represents a content download to be performed by an application.
 * Requests of this type are handled totally by an application. They are
 * created in the PENDING_NO_CONFLICT_STATE. The application handling the
 * request is responsible for downloading the content and all resulting
 * state changes.<p>
 * Requests of this type shall always be visible to the application specified
 * when the corresponding ApplicationDownloadSpec was created regardless of
 * recording request specific security attributes.
 * The methods defined in this class shall always fail with a ApplicationDownloadException
 * if called by any application other than the one specified when
 * the corresponding ApplicationDownloadSpec was created.
 */
public interface ApplicationDownloadRequest
   extends org.ocap.shared.dvr.LeafRecordingRequest {
     /**
      * Return a file to which downloaded data can be written and from
      * which downloaded data can be read.
      * @return a File
      * @throws ApplicationDownloadException if the caller is not permitted
      * to call this method as defined above
      */
     RandomAccessFile getFile() throws ApplicationDownloadException;
     /**
      * Set the state of the download.
      * @param state the new state of the recording
      * @throws ApplicationDownloadException if the caller is not permitted
      * to call this method as defined above
      */
     public void setState( int state );

     /**
      * Set the reason to use for the exception returned by getFailedException.
      * The reason must be one valid for the constructor of
      * RecordingFailedException.
      * @param reason the reason to use when constructing a
      * RecordingFailedException
      */
     public void setFailedReason( int reason );
}
```

Class LocatorDownloadSpec

```
package org.oipf.download;

import javax.tv.service.selection.InvalidServiceComponentException;
import javax.tv.locator.Locator;
import java.util.Date;
import org.ocap.shared.dvr.RecordingProperties;
```



```
/**
 * Specifies a recording request in terms of a Locator to a file.
 * The identified file is asynchronously downloaded from the network.<p>
 * When instances of this class are passed to RecordingManager.record(..),
 * the following additional failure mode shall apply - if the locator
 * does not reference a file then the record method shall throw an
 * IllegalArgumentException.<p>
 * No additional failure modes are defined for RecordingRequest.reschedule.<p>
 * For recording requests resulting from a recording spec of this type,
 * downloading shall start immediately. Such recording requests will never be
 * in a pending or a IN_PROGRESS_INCOMPLETE_STATE state.
 */

public class LocatorDownloadSpec extends org.ocap.shared.dvr.RecordingSpec
{
     /**
      * Constructor
      * @param source Source of content to be downloaded
      * @param properties the definition of how the recording is to be done
      */
     public LocatorDownloadSpec(Locator source, RecordingProperties properties)
          { super(properties); }

     /**
      * Returns the source of the content to be downloaded
      * @return the source passed into the constructor
      */
     public Locator getSource()
          { return null; }
}
```

Class ApplicationDownloadException

```
package org.oipf.download;
/**
 * Thrown when an application calls methods on ApplicationDownloadRequest
 * which it is not permitted to call.
 **/
public class ApplicationDownloadException extends Exception {
     /**
      * Constructs a <code>ApplicationDownloadException</code> with <code>null</code>
      * as its error detail message.
      */
     public ApplicationDownloadException() { super();}

     /**
      * Constructs a <code>ApplicationDownloadException</code> with the specified detail
      * message. The error message string <code>s</code> can later be
      * retrieved by the <code>{@link java.lang.Throwable#getMessage}</code>
      * method of class <code>java.lang.Throwable</code>.
      *
      * @param   s   the detail message.
      */
     public ApplicationDownloadException(String s) { super(s); }
}
```


Class ApplicationDownloadSpec

```
package org.oipf.download;
import org.dvb.application.AppID;
import org.ocap.shared.dvr.RecordingProperties;
/**
 * Represents a content download to be performed by an application.
 * The application to handle the request and the source of the content
 * to download must be specified when the request is created.
 * No other application is permitted to handle the request.
 * The format of the source must be co-ordinated between the application
 * requesting the download and the application handling the download request.
 */
public class ApplicationDownloadSpec extends org.ocap.shared.dvr.RecordingSpec {
     /**
      * Create a request for content download by application.
      * @param app the application to perform the download
      * @param source the source of the download
      * @param properties the properties for the download
      */
     public ApplicationDownloadSpec(AppID app, String source,
                     RecordingProperties properties)
          {super(properties);}
     /**
      * Return the application ID specified.
      * @return an application ID
      */
     public AppID getAppID(){return null;}
     /**
      * Return the source specified.
      * @return a String
      */
     public String getSource(){return null;}
}
```


Appendix D. Package org.oipf.service

Interface ServiceCreator

```
package org.oipf.service;
import javax.tv.locator.Locator;
import javax.tv.service.Service;
import javax.tv.locator.InvalidLocatorException;
/**
 * This interface extends javax.tv.service.SIManager with overloaded methods
 * for creating IPTV services.
 */
public interface ServiceCreator {
     /**
      * Get a service from a locator for the streaming component
      * and an XML AIT fragment for the application component.
      * This permits service bound applications to be used without
      * application signalling in-band in the A/V content.
      * @param locator A locator identifying a service
      * @param ait An XML AIT fragment
      * @return a Service
      * @throws InvalidLocatorException If locator does not identify a valid
      * Service.
      * java.lang.SecurityException If the caller does not have
      * javax.tv.service.ReadPermission(locator).
      */
     public abstract Service getService( Locator locator, String ait );

     /**
      * Get a service from a piece of SDP information describing some A/V
      * content.
      * @param sdp the piece of SDP information
      * @return a Service
      * @throws InvalidLocatorException If locator n the SDP information
      * does not identify a valid Service.
      * @throws java.lang.SecurityException If the caller does not have
      * javax.tv.service.ReadPermission for the locator in the SDP information
      */
     public abstract Service getService( String sdp );

     /**
      * Get a service from a piece of SDP information describing some A/V content and
      * an XML AIT fragment for the application component. This permits service bound
      * applications to be used without application signalling in-band in the A/V
      * content.
      * @param sdp the piece of SDP information
      * @param ait An XML AIT fragment
      * @return a Service
      * @throws InvalidLocatorException If locator n the SDP information does not
      * identify a valid Service.
      * @throws java.lang.SecurityException If the caller does not have
      * javax.tv.service.ReadPermission for the locator in the SDP information
      */
     public abstract Service getService( String sdp, String ait );
}
```


Appendix E. org.oipf.auth

Class HTTPDigestCredentials

```
package org.oipf.auth;
/**
 * Credentials to be used
 */
public class HTTPDigestCredentials extends UserCredentials
{
      /**
       * Create an HTTPDigestCredentials.
       * @param domain the domain
       * @param realm the realm
       * @param username the username
       * @param password the password
       */
      public HTTPDigestCredentials( String domain, String realm, String username,
            String password ) {super(domain);}
      /**
       *
       */
      public String getRealm(){return null;}
      /**
       *
       */
      public String getUsername() {return null;}
      /**
       *
       */
      public String getPassword() { return null;}

}
```

Class UserAuthenticationPermission

```
package org.oipf.auth;
/**
 * This class represents a Permission to access user authentication information
 * for a particular domain.
 * The name string is a fully qualified domain name. The actions string is unused.
 */
public final class UserAuthenticationPermission extends java.security.BasicPermission {
      /**
       * Creates a new UserAuthenticationPermission.
       */
      public UserAuthenticationPermission() {super("toto");}

      /**
       * Creates a new UserAuthenticationPermission.
       * The actions string is currently unused and should be null.
       * The name string shall be a fully qualified domain name.
       * @param name the name of the permission
       * @param actions the actions string
       */
```



```
      public UserAuthenticationPermission(String name, String actions)
            {super(name);}

      /**
       * Returns the list of actions that had been passed to the
       * constructor - it shall return null.
       *
       * @return a null String.
       */
      public String getActions() { return null;}

      /**
       * Checks if this UserAuthenticationPermission object "implies" the
       * specified permission.
       * @param permission the specified permission to check.
       * @return true if and only if the specified permission is an instanceof
       * and the domain in this object domain-matches
       * the domain in the specified permission as defined by RFC2965.
       */
      public boolean implies(java.security.Permission permission) {return true;}

      /**
       * Checks for equality against this UserAuthenticationPermission object.
       * @param obj the object to test for equality with this
       * UserAuthenticationPermission object.
       * @return true if and only if obj is an UserAuthenticationPermission and
       * the two domain names are equal.
       */
      public boolean equals(Object obj) { return true;}

      /**
       * Returns the hash code value for this object.
       * @return the hash code value for this object.
       */
      public int hashCode() { return 0 ;}
}
```

Class UserAuthenticationManager

```
package org.oipf.auth;
/**
 *
 */
public class UserAuthenticationManager {
      /*
       */
      private UserAuthenticationManager() {}

      /**
       * Return a UserAuthenticationManager
       * @return a UserAuthenticationManager
       */
      public static UserAuthenticationManager getUserAuthenticationManager()
            {return null;}
```



```
     /**
      * Get credentials which authenticate the current user to the specified
      * domain.
      * @param domain the domain
      * @return an array of credentials objects
      * @throws SecurityException if the calling application does not have
      * UserAuthenticationPermission for the domain
      */
     public UserCredentials[] getCredentials( String domain ) { return null;}

     /**
      * Set credentials
      * @param credentials the credentials to set
      * @throws SecurityException if the calling application is not permitted
      * to set credentials for the domain
      */
     public void setCredentials( UserCredentials credentials ){}
}
```

Class UserCredentials

```
package org.oipf.auth;
/**
 * Base class for different kinds of credentials
 */
public abstract class UserCredentials
{
     /**
      * Create a UserCredentials
      * @param domain the domain to which the credentials apply
      */
     public UserCredentials( String domain ){}
     /*
      * Return the domain for which a set of credentials apply
      * @return a domain
      */
     public String getDomain() { return null ; }
}
```

Class CookieCredentials

```
package org.oipf.auth;
import java.util.Date;
/**
 * Represents a HTTP Cookie.
 * This class is purely a container for the information passed in
 * through the constructor. No checking of the information is
 * performed. The accessor methods return the values passed in
 * through the constructor.
 */
public class CookieCredentials extends UserCredentials
{
     /**
      * Construct a cookie
      * @param domain the domain to which the cookie is sent
      * @param path the path within the domain
```



```
     * @param name the name of the cookie
     * @param value the value of the cookie
     * @param expiry the expiry date of the cookie or null if none is set
     */
    public CookieCredentials( String domain, String path, String name,
              String value, Date expiry )
         {super(domain); }
    /**
     * return the name of the cookie
     * @return the name as passed to the constructor
     */
    public String getName() { return null;}
    /**
     * return the value of the cookie
     * @return the value as passed to the constructor
     */
    public String getValue() { return null;}
    /**
     * return the expiry date of the cookie
     * @return the expiry date of the cookie or null if none is set
     */
    public Date getExpiry() { return null;}
    /**
     * return the path of the cookie
     * @return the path as passed to the constructor
     */
    public String getPath() { return null;}
}
```


Appendix F. org.oipf.uiserver

Class UIServerManager

```
package org.oipf.uiserver;
/**
 * Enables applications to register as providing a remote UI.<p>
 * This class is intended to be used as follows;<ul>
 * <li>The application starts listening on one or more ports.
 * <li>For each service offered, the application constructs URLs
 * from the IP address of the device, the port number and any
 * path element used by the application.
 * <li>For each service offered, the application constructs XML
 * fragments as specified by CEA-2014.
 * <li>The remote UIs provided by all registered applications are combined
 * into a single remote UI listing as defined by CEA-2014.
 * <li>The results are made available to remote UI clients as defined
 * by CEA-2014.
 * </ul>
 */
public class UIServerManager {
      /*
       * Private constructor to stop javadoc generating one.
       */
      private UIServerManager() {}
      /**
       * Obtain a UIServerManager
       */
      public static UIServerManager getManager(){ return null;}
      /**
       * Register as providing a remote UI service.
       * @param desc the entry for the service or services available on
       * this port formatted according to CEA-2014.
       * @throws IllegalArgumentException if there are errors in the
       * descriptions provided
       */
      public void registerUIServer( String desc[]){}
      /**
       * Unregister from providing a remote UI service
       */
      public void unregisterUIServer(){}
}
```


Appendix G. org.oipf.drm

The DRM Systems names are defined as URNs with the DVB CA System ID (16 bit number) in there. A DRM System name shall be signaled by prefixing the decimal number format of CA_System_ID with "urn:dvb:casystemid:" as defined for the DRMSystemID attribute in [OIPF_META2]. Note that the decimal number format of CA_System_ID SHALL not have leading zeroes.

Class DRMAgentEvent

```
package org.oipf.drm;
/**
 * The <code>DRMAgentEvent</code> class reports events from
 * a DRM system.
 * @since   Release 1
 */
public class DRMAgentEvent extends java.util.EventObject{
     /**
      * ID indicating the cause of the event was
      * success of a message sent to a DRM agent.
      */
     static public final int SUCCESS = 0;
     /**
      * ID indicating the cause of the event was
      * failure of a message sent to a DRM agent
      */
     static public final int FAILURE = 1;
     /**
      * ID indicating the cause of the event
      * was something other than a message sent
      * to a DRM Agent using this package.
      */
     static public final int OTHER = 2;

     /**
      * Reason for failure events - cannot process request.
      * The request failed because the DRM agent was unable to
      * complete the necessary computations in the time allotted.
      */
     static public final int TIMEOUT = 1;
     /**
      * Reason for failure events - Unknown MIME type.
      * The request failed because the specified Mime Type is
      * unknown for this DRM system.
      */
     static public final int UNKNOWN_TYPE = 2;
     /**
      * Reason for failure events - User Consent Needed.
      * The request failed because user consent is needed for that action.
      */
     static public final int USER_CONSENT = 3;
     /**
      * Reason for failure events - other unspecified reason
      * The request failed because an unspecified error occurred.
      */
     static public final int UNSPECIFIED = 4;
```



```
     /**
      * Create a new DRMAgentEvent object.
      * <p>
      * @param cause ID indicating the cause of the event being sent
      * @param reason For FAILURE events, the reason for the failure, otherwise 0.
      * @param source the DRM agent which is the source of the event
      * @param msgID for SUCCESS or FAILURE events, the message id returned
      * when the corresponding message was sent to the DRM agent otherwise
      * null.
      */
     public DRMAgentEvent(DRMAgent source, int msgID, int cause, int reason )
          {super((Object)source);}

     /**
      * gets the message ID
      * @return the message ID as passed to the class constructor
      */
     public int getMsgID(){return 0;}

     /**
      * gets the cause of the event
      * @return one of the cause constants as defined in this class
      */
     public int getCause(){return 0;}

     /**
      * gets the reason for a failure
      * @return one of the reason constants as defined in this class
      */
     public int getReason(){return 0;}
}
```

Class DRMAgentPermission

```
package org.oipf.drm;

/**
 * This class represents a Permission to exchange messages with a DRM system.
 * The name string can be either a comma separated list of DRM system names
 * or "*". The actions string is unused.
 */
public final class DRMAgentPermission extends java.security.BasicPermission {
     /**
      * Creates a new DRMAgentPermission.
      */
     public DRMAgentPermission()
     {super("toto");}

     /**
      * Creates a new DRMAgentPermission.
      * The actions string is currently unused and should be null. The name string
      * shall be either a comma separated list of DRM  * system names or "*".
      * @param name the name of the permission
      * @param actions the actions string
      */
     public DRMAgentPermission(String name, String actions) {super(name);}
```



```
     /**
      * Returns the list of actions that had been passed to the
      * constructor - it shall return null.
      *
      * @return a null String.
      */
     public String getActions() { return null;}

     /**
      * Checks if this DRMAgentPermission object "implies" the specified permission.
      * @param permission the specified permission to check.
      * @return true if and only if the specified permission is an instanceof
      * DRMAgentPermission
      */
     public boolean implies(java.security.Permission permission) {return true;}

     /**
      * Checks for equality against this DRMAgentPermission object.
      *
      * @param obj the object to test for equality with this DRMAgentPermission object.
      * @return true if and only if obj is an DRMAgentPermission
      */
     public boolean equals(Object obj) { return true;}

     /**
      * Returns the hash code value for this object.
      *
      * @return the hash code value for this object.
      */
     public int hashCode() { return 0;}
}
```

Interface DRMAgentListener

```
package org.oipf.drm;

import java.util.*;

/**
 * The <code>DRMAgentListener</code> class allows an
 * application to be told about asynchronous events relating
 * to a DRM system. When a message is sent to a DRM agent,
 * success is always reported asynchronously and errors which
 * cannot be detected immediately are also reported asynchronously.
 *
 * @since   Release 1
 */

public interface DRMAgentListener extends EventListener{
     /**
      * An operation triggered by a message sent through
      * the DRMAgent has succeeded.
      * @param resultMsg the DRM specific message result
      * @param msgID the message ID provided when the message
      * which triggered this operation was sent to the DRM agent
```



```
      */
     public void success(String resultMsg, String msgID );

     /**
      * An operation triggered by a message sent through the
      * DRMAgent has failed.
      * @param reason the reason for the failure
      * @param msgID the message ID provided when the message
      * which triggered this operation was sent to the DRM agent
      */
     public void failed( String reason, String msgID );

     /**
      * An event has happened unrelated to a message sent through
      * the DRMAgent.
      * @param evt the event
      */
     public void receive( DRMAgentEvent evt );
}
```

Class DRMAgent

```
package org.oipf.drm;
import java.io.IOException;
/**
 * Represents a generic DRM Agent.
 */
public class DRMAgent {
     /*
      * private constructor
      */
     private DRMAgent(){}
     /**
      * Obtain a DRM agent
      * @param name the name of the DRM system to obtain an agent for
      * @return a DRMAgent or null if the named DRM system is not supported
      * @throws SecurityException if the DRM system is supported but the
      * calling application does not have either DRMAgentPermission with
      * the name of the system or with "*"
      */
     public static DRMAgent getAgent(String name){ return new DRMAgent(); }

     /**
      * Obtain the list of supported DRM systems.
      * @return an array of DRM system names. If no DRM systems are supported
      * then an array of length zero shall be returned.
      */
     public static String[] getAgents() { return null ; }
     /**
      * Send a message to a DRM agent.
      * @param msgType A globally unique message type as defined by the DRM
      * system. For example:- application/vnd.marlin.drm.actiontoken+xml
      * (i.e. MIME-type of Marlin Action Token)
      * @param message The message to be provided to the underlying DRM agent
      * formatted according to the message type as indicated by msgType
      * @throws DRMAgentException If the message cannot be processed and this
```



```
       * can be determined immediately without any network access. For example,
       * a syntax error in the message.
       * @return an ID which SHALL be part of all events associated with this message.
       */
      public int sendMessage(String msgType, String message)
            throws DRMAgentException { return 0; }

      /**
       * Fetch a message from the network and send it to a DRM agent.
       * Failure to process the message is reported asynchronously.
       * @param message the URL of the message to fetch
       * @param msgID A unique ID to identify the message, to be provided as
       * part of all events posted as a consequence of this message
       * @throws java.io.IOException if the message cannot be fetched
       */
      public void fetchAndSendMessage(java.net.URL message, String msgID)
            throws IOException {}

      /**
       * Add a listener for DRMAgent events.
       * If the listener is already added then this method has no effect.
       * @param l the listener to add
       */
      public void addListener( DRMAgentListener l) {}

      /**
       * Remove a listener for DRMAgent events.
       * If the listener is not added then this method has no effect.
       * @param l the listener to remove
       */
      public void removeListener( DRMAgentListener l) {}
}
```

Class DRMAgentException

```
package org.oipf.drm;

/**
 * The <code>DRMAgent</code> exception is thrown if a
 * message to be sent to a DRM agent contains an error and this can
 * be immediately detected. For example, a syntax error.
 *
 * @since   Release 1
 */
public class DRMAgentException extends Exception {
      /**
       * Construct a DRMAgentException with no detail message
       */
      public DRMAgentException() {}

      /**
       * Construct a DRMAgentException with a detail message
       * @param s detail message
       */
      public DRMAgentException(String s) {}
}
```


Class DRMRightsErrorEvent

```
package org.oipf.drm;
/**
 * Represents an error event raised when a player
 * tries to play a protected content without a license
 * or with an invalid license
 */
public class DRMRightsErrorEvent extends ControllerErrorEvent {
     /**
      *  errorState valid values
      */
     static public final int NOLICENSE = 0;
     static public final int LICENSEINVALID = 1;

     protected Integer errorState;

     /**
      * The unique identifier of the protected content in the scope of the DRM
      * system that raises the error
      */
     protected String contentID;

     /**
      * The DRM System name
      */
     protected String DRMSystemName;

     /**
      * The rightsIssuerUrl optional element indicating the URL that
      * can be used to non-silently obtain the rights for the content
      * item currently being played for which this DRM error is generated
      */
     protected String rightsIssuerUrl;

     /**
      * Constructor
      */
     public DRMRightsErrorEvent (Controller from, String message, Integer errorState,
               String contentID, String DRMSystemID, String rightsIssuerUrl) {
          super(from, message);
     }

     /**
      * Gets the errorState value
      * @return errorState describing the current state (NOLICENSE or LICENSEINVALID).
      */
     public Integer getErrorState() {return errorState;}

     /**
      * Gets the contentID value
      * @return contentID of the content.
      */
     public Integer getContentID () {return contentID;}
```



```
    /**
     * Gets the DRM System name value
     * @return DRM System name describing the DRM System that generated
     * the error.
     */
    public String getDRMSystemID () {return DRMSystemID;}

    /**
     * Gets the rightsIssuerUrl value
     * @return rightsIssuerUrl for license retrieval.
     */
    public String getRightsIssuerUrl () {return rightsIssuerUrl;}
}
```


Appendix H. org.oipf.pvr

Class RecordingAccessPermissions

```
package org.oipf.pvr;

/**
 * This class encapsulates permissions to access recordings.
 * World means all applications authorised to access persistent storage.
 * Owner means the application which created the file. Organisation is defined
 * as applications with the same organisation id as defined elsewhere in the
 * present document.
 * Additionally other organisation_ids may be permitted without providing world access.
 */

public class RecordingAccessPermissions extends
org.dvb.io.persistent.FileAccessPermissions {

     /**
      * This constructor encodes all the file access permissions as a set of booleans.
      *
      * @param readWorldAccessRight read access for all applications
      * @param writeWorldAccessRight write access for all applications
      * @param readOrganisationAccessRight read access for organisation
      * @param writeOrganisationAccessRight write access for organisation
      * @param readApplicationAccessRight read access for the owner
      * @param writeApplicationAccessRight write access for the owner
      * @param readWhiteList read access for applications from these organisations
      * @param writeWhiteList write access for applications from these organisations
      */
     public RecordingAccessPermissions(boolean readWorldAccessRight,
               boolean writeWorldAccessRight,
               boolean readOrganisationAccessRight,
               boolean writeOrganisationAccessRight,
               boolean readApplicationAccessRight,
               boolean writeApplicationAccessRight,
               int[] readWhiteList, int[] writeWhiteList)
          { super(readWorldAccessRight, writeWorldAccessRight,
          readOrganisationAccessRight, writeOrganisationAccessRight,
          readApplicationAccessRight, writeApplicationAccessRight); }

     /**
      * Query whether this permission includes read access for the organisation
      * @param org_id the organisation
      * @return true if applications in this organisation can have read access,
      *  otherwise false.
      */
     public boolean hasReadOrganisationAccessRight(int org_id) {return false;}

     /**
      * Query whether this permission includes write access for the organisation
      * @param org_id the organisation
      * @return true if applications in this organisation can have read access,
      *  otherwise false.
      */
```



```
     public boolean hasWriteOrganisationAccessRight(int org_id) {return false;}
}
```

Class RecordingProperties

```
package org.oipf.pvr;


/**
 * Encapsulates the details about how a recording is to be made.
 */

public class RecordingProperties extends org.ocap.shared.dvr.RecordingProperties {
     /**
      * Constructor
      * @param expirationPeriod the period in seconds after the initiation of
      * recording when leaf recording requests with this recording
      * property are deemed as expired.
      * @param validityPeriod the period in seconds after the registration of the
      * RecordingRequest before which recordings which have not been completed
      * successfully shall not be discarded.
      * @param requestedRecording the recording access permissions for recording
      * requests
      * @param completedRecording the recording access permissions for completed
      * recordings
      * @param priority the priority of this recording request
      * @throws IllegalArgumentException if the expirationPeriod or the validityPeriod
               * is negative
      */
     public RecordingProperties(long expirationPeriod,
          long validityPeriod, RecordingAccessPermissions requestedRecording,
          RecordingAccessPermissions completedRecording, int priority)
          throws IllegalArgumentException
     {
          super(expirationPeriod);
     }
     /**
      * Return the period in seconds after the registration of
      * the <code>RecordingRequest</code> after which recordings which have
      * not been completed successfully should be discarded.
      * Implementations may impose a upper bound on the validity period and
      * cap the validity period requested to that upper bound.
      * @return the validity period that will be used for the recording
      * after capping to any implementation defined upper bound
      */
     public long getValidityPeriod() {return 0;}
     /**
      * Return the recording access permissions for recording requests.
      * @return the recording access permissions for recording requests
      */
     public RecordingAccessPermissions getRequestedRecordingPermissions(){return null;}
     /**
      * Return the recording access permissions for completed recordings
      * @return the recording access permissions for completed recordings
      */
```



```
     public RecordingAccessPermissions getCompletedRecordingPermissions()
          {return null;}
     /**
      * Return the priority for this recording request.
      * @return the priority for this recording request.
      */
     public int getPriority(){return 0;}
}
```

