International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

J.296
(06/2012)

SERIES J: CABLE NETWORKS AND TRANSMISSION OF TELEVISION, SOUND PROGRAMME AND OTHER MULTIMEDIA SIGNALS

Cable modems

Specifications for a hybrid cable set-top box

Recommendation ITU-T J.296

Recommendation ITU-T J.296

Specifications for a hybrid cable set-top box

Summary

Recommendation ITU-T J.296 describes the specifications for a hybrid cable set-top box (STB) to be used in cable TV networks. The hybrid cable set-top box, called the hybrid STB in the cable television network environment, is a key device for the reception of broadcast services and Internet Protocol (IP) based services. By combining a cable platform and a home network, the STB can offer new interactive services to cable customers and cable operators. This Recommendation contains descriptions of STB architecture, hardware, software, media-processing, and applications for a hybrid cable STB.

History

Edition	Recommendation	Approval	Study Group
1.0	ITU-T J.296	2012-06-29	9

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

Table of Contents

Page

Page

Recommendation ITU-T J.296

Specifications for a hybrid cable set-top box

1 Scope

This Recommendation describes the specifications for a hybrid cable set-top box (STB) to be used in cable TV networks. The hybrid cable set-top box, called the hybrid STB in the cable television network environment, is a key device for the reception of broadcast services and Internet Protocol (IP) based services. By combining a cable platform and a home network, the STB can offer new interactive services to cable customers and cable operators. This Recommendation contains descriptions of STB architecture, hardware, software, media-processing, and applications for a hybrid cable STB.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T F.902] Recommendation ITU-T F.902 (1995), *Interactive services design guidelines*.

[ITU-T H.262] Recommendation ITU-T H.262 (2012)) | ISO/IEC 13818-2: 2013, *Information technology – Generic coding of moving pictures and associated audio information: Video*.

[ITU-T H.264] Recommendation ITU-T H.264 | International Standard ISO/IEC 14496-10 (2010), *Advanced video coding for generic audiovisual services*.

[ITU-T H.741.0] Recommendation ITU-T H.741.0 (2011), *IPTV application event handling: Overall aspects of audience measurement for IPTV services*.

[ITU-T J.90] Recommendation ITU-T J.90 (2000), *Electronic programme guides for delivery by digital cable television and similar methods – Reference operating scenario and requirements*.

[ITU-T J.93] Recommendation ITU-T J.93 (1998), *Requirements for conditional access in the secondary distribution of digital television on cable television systems*.

[ITU-T J.94] Recommendation ITU-T J.94 (1998), *Service information for digital broadcasting in cable television systems*.

[ITU-T J.112] Recommendation ITU-T J.112 (1998), *Transmission systems for interactive cable television services*.

[ITU-T J.122] Recommendation ITU-T J.122 (2007), *Second-generation transmission systems for interactive cable television services – IP cable modems*.

[ITU-T J.142] Recommendation ITU-T J.142 (2000), *Methods for the measurement of parameters in the transmission of digital cable television signals*.

[ITU-T J.160] Recommendation ITU-T J.160 (2005), *Architectural framework for the delivery of time-critical services over cable television networks using cable modems*.

[ITU-T J.200] Recommendation ITU-T J.200 (2010), *Worldwide common core – Application environment for digital interactive television services*.

[ITU-T J.201] Recommendation ITU-T J.201 (2009), *Harmonization of declarative content format for interactive television applications*.

[ITU-T J.202] Recommendation ITU-T J.202 (2010), *Harmonization of procedural content formats for interactive TV applications*.

[ITU-T J.222.0] Recommendation ITU-T J.222.0 (2007), *Third-generation transmission systems for interactive cable television services – IP cable modems: Overview*.

[ITU-T J.222.1] Recommendation ITU-T J.222.1 (2007), *Third-generation transmission systems for interactive cable television services – IP cable modems: Physical layer specification*.

[ITU-T J.222.2] Recommendation ITU-T J.222.2 (2007), *Third-generation transmission systems for interactive cable television services – IP cable modems: MAC and Upper Layer protocols*.

[ITU-T J.222.3] Recommendation ITU-T J.222.3 (2007), *Third-generation transmission systems for interactive cable television services – IP cable modems: Security services*.

[ITU-T J.295] Recommendation ITU-T J.295 (2012), *Functional requirements for a hybrid cable set-top box*.

[ITU-T L.1000] Recommendation ITU-T L.1000 (2011), *Universal power adapter and charger solution for mobile terminals and other hand-held ICT devices*.

[ITU-T Q.1741.1] ITU-T Recommendation Q.1741.1 (2002), *IMT-2000 references to release 1999 of GSM evolved UMTS core network with UTRAN access network.*

[ITU-T X.1191] Recommendation ITU-T X.1191 (2009), *Functional requirements and architecture for IPTV security aspects*.

[ITU-T X.1192] Recommendation ITU-T X.1192 (2011), *Functional requirements and mechanisms for the secure transcoding of IPTV.*

[ITU-T X.1193] Recommendation ITU-T X.1193 (2011), *Key management framework for secure Internet protocol television (IPTV) services*.

[ITU-T X.1195] Recommendation ITU-T X.1195 (2011), *Service and content protection interoperability scheme*.

[ITU-T Y.1910] Recommendation ITU-T Y.1910 (2008), *IPTV functional architecture*.

[ITU-T Y.2016] Recommendation ITU-T Y.2016 (2009), *Functional requirements and architecture of the NGN for applications and services using tag-based identification*.

[ITU-T Y.2111] Recommendation ITU-T Y.2111 (2011), *Resource and admission control functions in next generation networks*.

[ITU-R BT.1306-6] Recommendation ITU-R BT.1306-6 (2011), *Error correction, data framing, modulation and emission methods for digital terrestrial television broadcasting*.

[ITU-R BT.1699] Recommendation ITU-R BT.1699 (2002), *Harmonization of declarative application formats for interactive TV.*

[ITU-R BT.1700] Recommendation ITU-R BT.1700 (2002), *Characteristics of composite video signals for conventional analogue television systems*.

[IEC 60958] IEC 60958 (2008), *Digital audio interface – Part 1: General*.

[IEC 61937-4] IEC 61937-4 (2003), *Digital audio – Interface for non-linear PCM encoded audio bitstreams applying IEC 60958 – Part 4: Non-linear PCM bitstreams according to the MPEG audio format*.

[IEC 61937-6] IEC 61937-6 (2006), *Digital audio – Interface for non-linear PCM encoded audio bitstreams applying IEC 60958 – Part 6: Non-linear PCM bitstreams according to the MPEG-2 AAC and MPEG-4 AAC audio formats*.

[IEC 62481-1] IEC 62481-1 (2007), *Digital living network alliance (DLNA) home networked device interoperability guidelines – Part 1: Architecture and protocols*.

[IEC 62481-2] IEC 62481-2 (2007), *Digital living network alliance (DLNA) home networked device interoperability guidelines – Part 2: DLNA media formats*.

[IEC 62546] IEC 62546 (2009), *High Definition (HD) recording link guidelines*.

[ISO/IEC 7816-15] ISO/IEC 7816-15:2004, *Identification cards – Integrated circuit cards – Part 15: Cryptographic information application*.

[ISO/IEC 10918-1] ISO/IEC 10918-1:1994, *Information technology – Digital compression and coding of continuous-tone still images: Requirements and guidelines (1994/02).*

[ISO/IEC 11172] ISO/IEC 11172:1993, *Information technology – Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s – Part 1: Systems (1993/08).*

[ISO/IEC 13818-3] ISO/IEC 13818-3:1998, *Information technology – Generic coding of moving pictures and associated audio information – Part 3: Audio*.

[ISO/IEC 13818-7] ISO/IEC 13818-7:2010, *Corrigendum 2 – Information technology – Generic coding of moving pictures and associated audio information – Part 7: Advanced Audio Coding (AAC).*

[ISO/IEC 14496-2] ISO/IEC 14496-2:2004, *Information technology – Coding of audio-visual objects – Part 2: Visual*.

[ISO/IEC 14496-3] ISO/IEC 14496-3:2009, *Information technology – Coding of audio-visual objects – Part 3: Audio).*

[ISO/IEC 14496-10] ISO/IEC 14496-10, *Information technology – Coding of audio-visual objects – Part 10: Advanced Video Coding.*

[ISO/IEC 14496-12] ISO/IEC 14496-12:2011, *Corrigendum 4 – Information technology – Coding of audio-visual objects – Part 12: ISO base media file format.*

[ISO/IEC 14496-14] ISO/IEC 14496-14:2010, *Amendment 1 – Information technology – Coding of audio-visual objects – Part 14: MP4 file format – Handling of MPEG-4 audio enhancement layers*.

[ISO/IEC 14496-15] ISO/IEC 14496-15:2010, *Information technology – Coding of audio-visual objects – Part 15: Advanced Video Coding (AVC) file format*.

[IEEE 802.11] IEEE 802.11-2007, *Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*.

[IETF RFC 791] IETF RFC 791 (1981), *Internet Protocol.*

[IETF RFC 793] IETF RFC 793 (1981), *Transmission Control Protocol.*

[IETF RFC 2131] IETF RFC 2131 (1997), *Dynamic Host Configuration Protocol.*

[IETF RFC 2616] IETF RFC 2616 (1999), *Hypertext Transfer Protocol – HTTP/1.1.*

[IETF RFC 3315] IETF RFC 3315 (2003), *Dynamic Host Configuration Protocol for IPv6 (DHCPv6).*

[IEEE RFC 4862] IETF RFC 4862 (2007), *IPv6 Stateless Address Autoconfiguration.*

[BBF TR-069] Broadband Forum TR-069 (2007), *CPE WAN Management Protocol v1.1.*

[BBF TR-106] Broadband Forum TR-106 Issue: 1 Amendment 6 (2011), *Data Model Template for TR-069-Enabled Devices.*

[BBF TR-135] Broadband Forum TR-135 (2007), *Data Model for TR-069 Enabled STB Version 1.1.*

[BBF TR-140] Broadband Forum TR-140 Amendment 1 (2010), *Data Model for Storage Service Enabled Device.*
http://www.broadband-forum.org/technical/download/TR-140_Amendment-1.pdf

[BBF TR-157] Broadband Forum TR-157 (2009), *Component Objects for CWMP.*

[JLabs E-023] Japan Cable Laboratories SPEC E-023 (2012), *Specification of hybrid cable set-top box.*
http://www.jlabs.or.jp/images/stories/PDF/jlabs_spec-e-023_ver.1.1.pdf

3 Terms and definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 cable modem (CM) [ITU-T J.112]: A modulator-demodulator at subscriber locations intended for use in conveying data communications on a cable television system.

3.1.2 cable television [ITU-T J.142]: Communications systems distributes broadcast and non-broadcast signals, as well as a multiplicity of satellite signals originating programming and other signals by means of coaxial cable and/or optical fibre.

3.1.3 conditional access system [ITU-T X.1193]: A component of a service and content protection system, the purpose of which is to prevent unauthorized (unentitled) access to a service or to content.

3.1.4 configuration package [ITU-T H.741.0]: A configuration package is the data structure which specifies the target services to be measured, content filtering, measurement schedule, events and samples to be measured, and measurement report delivery.

3.1.5 electronic programme guide [ITU-T J.90]: A structured multimedia database, intended to provide information on programmes to be broadcast or cablecast.

3.1.6 entitlement management messages (EMM) [ITU-T J.94]: Private conditional access information which specifies the authorization levels or the services of specific decoders; they may be addressed to an individual decoder or groups of decoders.

3.1.7 Hypertext Transfer Protocol (HTTP) (based on [IETF RFC 2616]): An application-layer protocol used to transmit data over the World-Wide Web.

3.1.8 identifier [ITU-T Y.2016]: An identifier is a series of digits, characters and symbols or any other form of data used to identify subscriber(s), user(s), network element(s), function(s), network entity(ies) providing services/applications, or other entities (e.g., physical or logical objects). Identifiers can be used for registration or authorization.

NOTE – Identifier can be either public to all networks, shared between a limited number of networks or private to a specific network (private identifiers are normally not disclosed to third parties).

3.1.9 measurement report [ITU-T H.741.0]: The data that the audience measurement function (AMF) generates from an end-user behaviour event or sample.

3.1.10 network [ITU-T J.94]: A collection of MPEG-2 Transport Stream (TS) multiplexes transmitted on a single delivery system, e.g. all digital channels on a specific cable system.

3.1.11 network address translator (NAT) [ITU-T Y.2111]: An entity that implements network address translation or NAPT functions. It consists of two types of NATs: near-end NAT that can be controlled by the operators directly, and far-end (remote) NAT that cannot be controlled by the operators directly.

3.1.12 pay-per-view [ITU-T J.93]: A payment system whereby the subscriber can pay for an individual program or specified period of time.

3.1.13 protocol [ITU-T J.112]: A set of rules and formats that determines the communication behaviour of layer entities in the performance of the layer functions.

3.1.14 proxy [ITU-T J.160]: A facility that indirectly provides some service or acts as a representative in delivering information, thereby relieving a host from having to support the services itself.

3.1.15 quality of service (QoS) [ITU-T Q.1741.1]: The collective effect of service performances which determine the degree of satisfaction of a user of a service. It is characterized by the combined aspects of performance factors applicable to all services, such as:

- Service operability performance;
- Service accessibility performance;
- Service retain ability performance;
- Service integrity performance; and
- Other factors specific to each service.

3.1.16 service information (SI) [ITU-T J.94]: Digital data describing the delivery system, content, and scheduling/timing of broadcast data streams etc. It includes MPEG-2 Program Specific Information (PSI) together with independently defined extensions.

3.1.17 uniform resource identifier (URI) [ITU-T J.200]: An addressing method to access a resource in local storage or on the Internet.

3.1.18 uniform resource locator (URL) [ITU-T M.3030]: An IETF standard. It describes the location of Web resources; a hierarchical scheme consisting of a protocol (e.g., http), followed by a hostname (e.g., www), and then a datapath.

3.1.19 user interface [ITU-T F.902]: Software and hardware components through which a user can interact with a system.

3.1.20 video on demand (VoD) [ITU-T Y.1910]: A service in which the end user can, on demand, select and view video content and where the end user can control the temporal order in which the video content is viewed (e.g., the ability to start the viewing, pause, fast forward, rewind, etc.).

NOTE – The viewing may occur some time after the selection of the video content.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 Advanced Audio Coding (AAC) (based on [ISO/IEC 13818-7]): An audio coding system standardized by the International Organization for Standardization.

3.2.2 audience measurement: The measurement of user viewing conditions, records of viewing, recording, playing, and manipulation, based on data collected with the permission of users for improvement of personalized services.

3.2.3 audience measurement functions: The functions that, when given permission, measure end-user behaviour by processing events or samples from cable TV services. Such functions may request and collect end-user information. They transfer processed events, samples, and end-user information to aggregation functions.

3.2.4 Audio Interchange File Format (AIFF) (based on [b-DAVIC 1.4.1]): The Audio Interchange File Format is defined in [b-DAVIC 1.4.1] Specification Part 9 Annex B; it is the file format for audio encoding of pulse code modulation (PCM).

3.2.5 broadcast transport stream: The broadcast transport stream is composed of MPEG-2 transport stream (TS) signals, which have multi-frame architecture using the same frame length as the orthogonal frequency-division multiplexing (OFDM) frame.

3.2.6 CA_system_id: The conditional access system identifier having a 16-bit field, which can identify the conditional access (CA) system.

3.2.7 channel number: The channel number refers to the numbers specified by broadcasting stations to label them (1 to 12). These numbers should correspond to the number of one-touch buttons on a remote controller unit.

3.2.8 channel scan (scan): The channel scan function searches all channels to determine the presence or absence of broadcast signals. It is used by viewers when it is unknown which terrestrial broadcasting programmes can be received in the area of residence. When a broadcast signal is detected, the information on the station will be registered.

3.2.9 copy control: A control copy generation that limits copying when the programme and other copyright objects are copied by the recording equipment connected with the broadcasting receiver.

3.2.10 Data-Over-Cable Service Interface Specification (DOCSIS): The cable modem standard that was established by the Multimedia Cable Network System Partners, Ltd. (MCNS) in 1997. This term is generally used as a generic term for a cable modem and is implemented in [ITU-T J.122] or [ITU-T J.112].

3.2.11 declarative application environment: An application environment using HTML.

3.2.12 domain name system (DNS) (based on [IETF RFC 1034]): A protocol used by the service that maps a host name on a network onto its IP address.

3.2.13 Dynamic Host Configuration Protocol (DHCP) (based on [IETF RFC 2131]): A protocol used to automatically configure terminals on a TCP/IP network. For example, this protocol allows IP addresses to be assigned dynamically.

3.2.14 elementary stream (ES) (based on [ITU-T J.200]): A basic stream that contains video data, audio data, or private data. A single elementary stream is carried in a sequence of PES packets with one and only one stream.

3.2.15 emergency warning system: An emergency warning system is used for disaster broadcasts. Its startcontrol signal forces receiver units to receive the disaster broadcast programme.

3.2.16 engineering TS: The engineering transport stream (TS) is a kind of TS signal that transmits downloaded software contents or watching control information on a satellite broadcast network.

3.2.17 graphic interchange format (GIF): A bitmap image format that was introduced by CompuServe in 1987 and has since come into widespread usage on the World Wide Web due to its wide support and portability. GIF images are compressed using the Lempel-Ziv-Welch (LZW) lossless data compression technique to reduce the file size without degrading the visual quality.

3.2.18 graphical user interface: A graphical user interface is equipped with graphical input and output on the computer monitor.

3.2.19 hybrid cable set-box top (STB): An STB that satisfies the requirements defined in this Recommendation. Unless specifically noted otherwise, in this Recommendation the term STB shall mean the hybrid cable STB.

3.2.20 interleave frame allocation: A method of 3D video data for both-eye viewing.

3.2.21 Internet Protocol (IP) (based on [IETF RFC 791]): A network layer protocol that defines the addressing mechanism on the Internet to allow data to be transmitted.

3.2.22 Joint Photographic Experts Group (JPEG) (based on [ISO/IEC 10918]): JPEG is a standard format for compressing pictures. The degree of compression can be adjusted, allowing a selectable tradeoff between storage size and image quality.

3.2.23 JavaScript: JavaScript is a scripting language for a Web browser. JavaScript is an implementation of the ECMAScript language standard and is primarily used in the form of client-side JavaScript, implemented as part of a Web browser in order to provide enhanced-user interfaces and dynamic websites.

3.2.24 legacy application: An application or applications to support legacy services if the cable operator wishes.

3.2.25 management information base: A virtual database used for managing the entities in a communications network. Most often associated with the Simple Network Management Protocol (SNMP), the term is also used more generically in contexts such as in the OSI/ISO network management model.

3.2.26 middleware: Software within the STB that provides a set of application programming interfaces (APIs) against which applications can be developed, and that provide access to the resources and services of the STB.

3.2.27 Moving Pictures Expert Group-1 (MPEG-1) (based on [ISO/IEC 11172]): MPEG-1 is a data compression coding technology including video and audio, which is standardized by the International Organization for Standardization.

3.2.28 Moving Pictures Expert Group-2 (MPEG-2) (based on [ITU-T H.262] and [ITU-T J.94]): MPEG-2 is a compression and coding technology for data (such as moving images and audio data) specified by the International Organization for Standardization.

3.2.29 network address translation (based on [ITU-T Y.2111]): The operation by which IP addresses are translated (mapped) from one address domain to another address domain.

3.2.30 Network_id: The identifier assigned to each master transmitter.

3.2.31 network information table: The table that carries information to relate transmission path information, such as frequencies to channels, and that lists ID numbers for all the service channels contained in a distribution system.

3.2.32 open application: An application or applications that cable operators or subscribers can select and install, assuming downloadable applications in market.

3.1.33 orthogonal frequency division multiplexing (OFDM) (based on [ITU-R BT.1306-6]): A digital multi-carrier modulation scheme, which uses a large number of closely-spaced orthogonal sub-carriers.

3.2.34 parental control (viewer age restriction): A system to restrict programme viewing using a combination of an age restriction listed as a programme attribute and a parental level (minimum age for viewing) in the receiver, set by the user himself using a password.

3.2.35 platform: A business entity that manages and operates a collection service on a network of digital broadcasting.

3.2.36 portable network graphics: A graphics file format succeeding GIF. It is pronounced "PING" and is capable of lossless compression. The file format is comprised of an 8-byte signature followed by a series of "chunks".

3.2.37 portal site: An entrance web site of Internet, providing links, search engines, news, Web mail services, and electrical bulletin board, etc.

3.2.38 procedural application environment: An environment for necessary libraries for Java standard applications and broadcast co-ordinated Java applications.

3.2.39 remux: A method of transmission of TS created by the cable operator to cable a subscriber's STB over a cable TV network.

3.2.40 secure socket layer (SSL) (based on [ITU-R BT.1699]): A security protocol that works at a socket level. This layer exists between the TCP layer and the application layer to encrypt/decode data and authenticate concerned entities.

3.2.41 service:A set of functionalities, enabled by a provider for end users that provides, for example, a television programme delivery service, a content-on-demand service, IP connectivity, next generation services where the hybrid cable STB is associated with mobile devices, etc.

3.2.42 set-top box: A hardware box that contains a digital signal demodulator, de-multiplexer, decoder, and other functionalities and interfaces related to digital signal reception and presentation of the distributed programme at the subscriber's site. The set-top box that satisfies the requirements defined in this Recommendation is called a hybrid cable set-top box.

3.2.43 Simple Network Management Protocol (SNMP): An Internet-standard protocol for managing devices on IP networks. Essentially, SNMP agents expose management data on the managed systems as variables. The Protocol also permits active management tasks, such as modifying and applying a new configuration through remote modification of these variables. The variables that are accessible via SNMP are organized in hierarchies. These hierarchies, and other metadata (such as type and description of the variable), are described by management information bases (MIBs).

3.2.44 single-frequency network (SFN) (based on [ITU-R BT.1306-6]): A broadcast network where several transmitters simultaneously send the same signal over the same frequency channel.

3.2.45 standard application: An application which a cable operator uses for service delivery, to be installed before or after shipment on decision by the cable operator.

3.2.46 subtitle: A service of superimposing related text on a TV video broadcast.

3.2.47 superimpose: A subtitle provided asynchronously to the main video, audio, and data. It is used for up-to-the-minute news, changes in air times, and time signals.

3.2.48 terrestrial digital TV broadcast: A mode of digital television broadcasting which does not involve satellite transmission.

3.2.49 trans-modulation: A transmission method of a transport stream (TS). It is created and operated by other media and broadcasted to cable subscribers without changing parameters in the TS.

3.2.50 Transmission Control Protocol (TCP) (based on [IETF RFC 793]): A transport layer protocol that provides highly reliable end-to-end, connection-oriented data delivery using an error detection and correction mechanism.

3.2.51 transport stream (TS) (based on [ITU-T H.262]: The transport stream defined by the MPEG-2 system standard (in digital terrestrial television broadcasting, one TS is assigned to a master transmitter).

3.2.52 User Datagram Protocol (UDP) (based on [b-IETF RFC 768]): A transport layer protocol between two hosts without a confirmation function but which minimizes protocol overhead and is a connectionless type of communication suitable for services with high transmission efficiency.

3.2.53 view log: Information history of pay-per-view content viewing.

3.2.54 virtual machine (based on [ITU-T Q.1741.1]): A software program that simulates a hypothetical computer central processing unit. The programs executed by a virtual machine are represented as byte codes, which are primitive operations for this hypothetical computer.

3.3 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

3GPP	Third Generation Partnership Project
3GPP2	Third Generation Partnership Project 2
AAC	Advanced Audio Coding
ACS	Auto Configuration Server
AIFF	Audio Interchange File Format
AMF	Audience Measurement Function
AMR	Adaptive Multi-Rate
AP	Access Point
API	Application Programming Interface
ARIB	Association of Radio Industries and Businesses
AVC	Advanced Video Coding
BBF	Broadband Forum
BER	Bit Error Ratio
BML	Broadcast Markup Language
BS	Broadcasting Satellite
CA	Conditional Access
CAS	Conditional Access System
CATV	Community Antenna Television
CDN	Contents Delivery Network
CEC	Consumer Electronics Control
CGMS-A	Copy Generation Management System – Analogue
CM	Cable Modem
CPE	Customer Premises Equipment
CPU	Central Processing Unit

CS	Communications Satellite
CSS	Cascading Style Sheet
CWMP	CPE WAN Management Protocol
DAE	Declarative Application Environment
DHCP	Dynamic Host Configuration Protocol
DLNA	Digital Living Network Alliance
DMC	Digital Media Controller
DMIPS	Dhrystone Million Instructions Per Second
DMP	Digital Media Player
DMR	Digital Media Renderer
DMS	Digital Media Server
DNS	Domain Name System
DOCSIS	Data Over Cable Service Interface Specification
DoS	Denial of Service
DS	Downstream
DTCP-IP	Digital Transmission Content Protection over Internet Protocol
DVD	Digital Versatile Disc
DVR	Digital Video Recorder
ECG	Electronic Content Guide
EDID	Extended Display Identification Data
EMM	Entitlement Management Message
EPG	Electronic Programme Guide
ES	Elementary Stream
ESSID	Extended Service Set Identifier
FAT	File Allocation Table
FLUTE	File Delivery over Unidirectional Transport
FTTH	Fibre To The Home
GB	Gigabyte
GIF	Graphic Interchange Format
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HD	High Definition
HDCP	High-bandwidth Digital Content Protection system
HDD	Hard Disk Drive
HDMI	High-Definition Multimedia Interface
HE	Head End
HE-AAC	High-Efficiency Advanced Audio Coding

HFC	Hybrid Fibre-Coaxial
HLS	HTML Live Streaming
HP	High Profile
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol over Secure Socket Layer
HW	Hardware
I/F	Interface
IC	Integrated Circuit
ICT	Information and Communication Technologies
IEC	International Electrotechnical Commission
IGD	Internet Gateway Device
IP	Internet Protocol
IPTV	Internet Protocol Television
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
IR	Infrared
ISO	International Organization for Standardization
ISP	Internet Service Provider
JCTEA	Japan Cable Television Engineering Association
JLabs	Japan Cable Laboratories
JPEG	Joint Photographic Experts Group
L	Level
LAN	Local Access Network
LED	Light Emitting Diode
MAC	Media Access Control
MB	Megabyte
MIPS	Million Instructions Per Second
MP	Main Profile
MPEG	Moving Picture Expert Group
MVC	Multiview Video Coding
NAPT	Network Address Port Translation
NAS	Network Attached Storage
NIT	Network Information Table
NTSC	National Television System Committee
NVRAM	Non Volatile Random Access Memory
OFDM	Orthogonal Frequency Division Multiplexing

OIPF	Open IPTV Forum
OpenGL ES	Open Graphics Library for Embedded Systems
OpenGL	Open Graphics Library
OS	Operating System
OUI	Organizationally Unique Identifier
PC	Personal Computer
PCM	Pulse Code Modulation
PIN	Personal Identification Number
PPD	Pay Per Day
PPV	Pay Per View
PS	Program Stream
PVR	Personal Video Recorder
QAM	Quadrature Amplitude Modulation
RAM	Random Access Memory
RC	Reason Code
RCA	Radio Corporation of America
RCU	Remote Control Unit
RF	Radio Frequency
RF4CE	Radio Frequency for Consumer Electronics
RFC	Request For Comments
RJ	Registered Jack
ROM	Read Only Memory
RPC	Remote Procedure Call
SCMS	Serial Copy Management System
SCP	Service and Content Protection
SD	Standard Definition
SDP	Service Delivery Platform
SEI	Supplemental Enhancement Information
SFN	Single Frequency Network
SHP	Stereo High Profile
SI	Service Information
SNMP	Simple Network Management Protocol
SNR	Signal-to-Noise Ratio
SP	Simple Profile
SSDP	Simple Service Discovery Protocol
SSID	Service Set Identifier
SSL	Secure Socket Layer

STB	Set-Top Box
STUN	Simple Traversal of UDP through NATs
SW	Software
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TR	Technical Report
TS	Transport Stream
UDP	User Datagram Protocol
UI	User Interface
UPnP	Universal Plug and Play
URL	Uniform Resource Locator
US	Upstream
USB	Universal Serial Bus
VM	Virtual Machine
VOD	Video On Demand
W3C	World Wide Web Consortium
WAN	Wide Area Network
WEP	Wired Equivalent Privacy
WPA	Wi-Fi Protected Access
WPS	Wi-Fi Protected Setup

4 Conventions

In this Recommendation:

The keywords "**is required to**" indicate a requirement which must be strictly followed and from which no deviation is permitted if conformance to this document is to be claimed.

The keywords "**is recommended**" indicate a requirement which is recommended but which is not absolutely required. Thus this requirement need not be present to claim conformance.

The keywords "**is prohibited from**" indicate a requirement which must be strictly followed and from which no deviation is permitted if conformance to this document is to be claimed.

The keywords "**can optionally**" indicate an optional requirement which is permissible, without implying any sense of being recommended. This term is not intended to imply that the vendor's implementation must provide the option and the feature can be optionally enabled by the network operator/service provider. Rather, it means the vendor may optionally provide the feature and still claim conformance with the specification.

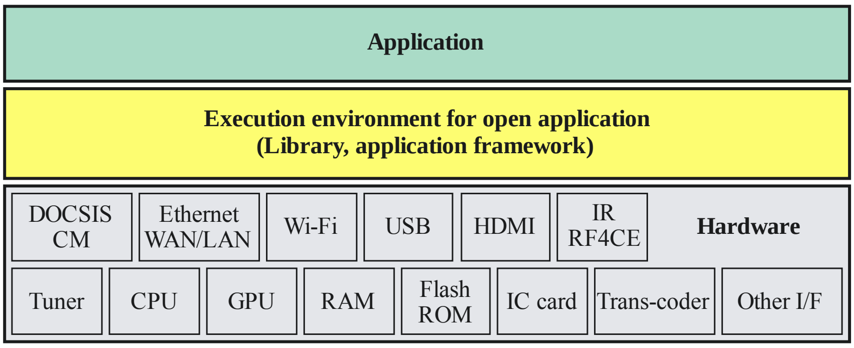
In the body of this document and its annexes, the words *shall*, *shall not*, *should*, and *may* sometimes appear, in which case they are to be interpreted, respectively, as *is required to*, *is prohibited from*, *is recommended*, and *can optionally*. The appearance of such phrases or keywords in an appendix or in material explicitly marked as *informative* are to be interpreted as having no normative intent.

5 Total configuration

5.1 Total configuration of the STB

This Recommendation specifies the overall architecture of the hybrid cable STB, consisting of the following functionalities:

- RF dual tuner
- IPTV signal receiving
- DOCSIS cable modem
- Communication interface (Ethernet, Wi-Fi)
- Additional functions (recording, Digital Living Network Alliance (DLNA), remote control, linking with mobile phone, etc.)
- External interface (universal serial bus (USB), high-definition multimedia interface (HDMI), conditional access integrated circuit (CA IC) card, remote control unit interface (RCU I/F), etc.)
- Software execution function (central processing unit (CPU), memory, graphics processing unit (GPU), operating system (OS), etc.)
- Open application execution environment (library, application framework, etc.).



J.296(12)_F5-1

Figure 5-1 – Overall functional diagram

The overall functional diagram of the STB is shown in Figure 5-1. The following items are implementation-dependent for the STB and outside the scope of this Recommendation.

- Internal storage device
- DVD and BD drivers
- Memory card slot
- High level personal authentication (such as picture (face), fingerprint, and voice recognition, etc.).

5.2 Software architecture of the STB

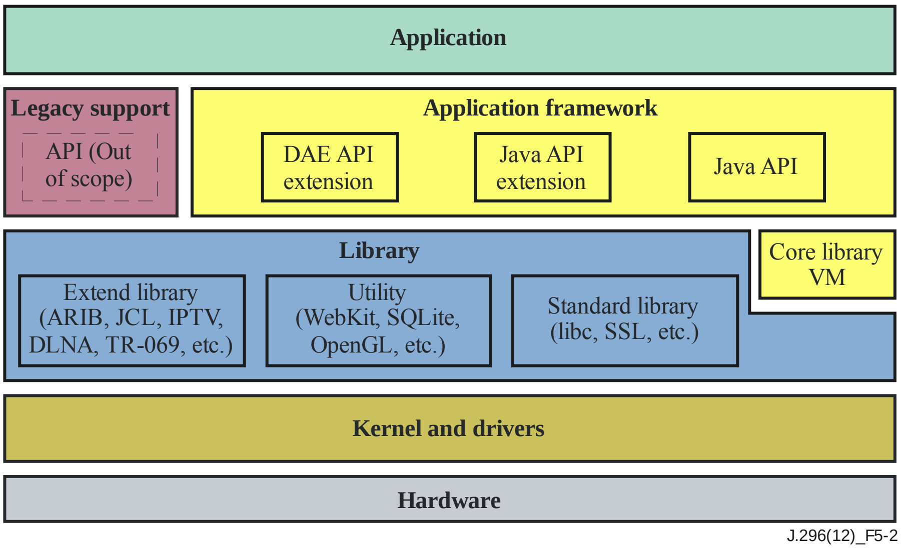


Figure 5-2 – Basic software architecture

Figure 5-2 shows the basic software architecture for the hybrid cable STB which contains the following elements:

- hardware
- operating system (OS) (kernel and driver)
- library (native layer middleware)
- application framework (virtual machine (VM) layer middleware)
- application.

The software portion of this architecture, (such as the OS, library, application framework, and application), is required to be upgradeable for security enhancement and updating with the latest ICT technology. The STB is required to support multi-tasking at the kernel level to support execution of several applications at the same time on the application framework. The STB is required to apply software architecture described in [JLabs E-023].

5.2.1 Library

Software that provides the basic functions of the STB is required to be implemented in the OS and library (native layer middleware). As the library on the STB, it is required to support the following functions:

Support of broadcasting and the cable television service

Examples of TV broadcasting, cable TV, and IPTV standards required by each region, are to be found in Table 5-1. See also clause 5.3.

Table 5-1 – Combination pattern for simultaneous usage

	Region A	Region B	Region C
TV broadcasting	DVB-T/T2/S/S2	ATSC	ARIB
Cable television	DVB-C/C2	SCTE,	JLabs, JCL, JCTEA
IPTV	DVB-IP	ATIS IIF	IPTV-FJ

– specifications in the declarative application environment

– specifications in the procedural application environment

– DLNA, Digital Transmission Content Protection over Internet Protocol, (DTCP-IP)

– device management (including remote control and provisioning).

Basic libraries

– Libraries for the declarative application environment: An HTML application environment, equipped with application programming interfaces (APIs) related to community antenna television (CATV) functionalities, is required. In addition, it is recommended to be equipped with the rendering engine supporting HTML5 and JavaScript for future enhancement.

– Libraries for the procedural application environment: A Java application environment for multimedia terminal devices, equipped with libc, webkit, SQLite, SSL, FreeType, Media Framework, SGL, OpenGL ES, Surface Manager, etc., is required.

– The STB can, upon request by the cable operator, optionally equip with extended libraries as implementation dependent.

Virtual machine

– A virtual machine for the above-described Java application environment is required.

5.2.2 Application framework (VM layer middleware)

The STB is required to support the execution and operation of the following applications:

– standard Java applications

– Java applications synchronized with broadcasting content

– next generation web applications.

The procedural application environment is defined in clause 9.4 as the execution environment of Java standard applications and Java applications synchronized with broadcast content. The hybrid cable STB is required to support the following API extensions:

– STB-specific feature control (channel selection, VoD, DVR control, etc.)

– control of broadcasting-related information (SI and PES acquisitions, etc.)

– transcoding (recording with further compression, transferring to other devices)

– encoding (optional, for communication application using USB devices, e.g., USB camera)

– interworking with in-home devices via DLNA.

5.2.3 Application

Various types of applications are executed on the above-mentioned application platforms supplied with the STB. The applications can be classified into three categories:

1) applications for basic STB features;

2) applications for cable television specific services; and

3) applications distributed in the open market.

These applications are required to be developed and implemented based on the API specifications defined in Appendices III and IV of this Recommendation. Appendix III describes the API extension in a declarative application environment (DAE), and Appendix IV is for the Java extended API in a procedural application environment.

This Recommendation defines the following three basic applications, which shall be selected by the cable operator:

– electronic programme guide (EPG) application

– application for recording functionalities, e.g., schedule recoding, playback of recorded programme, etc.

– STB installation support application, e.g., initial settings of broadcasting reception, network connection, etc.

This Recommendation does NOT define any requirements that apply to the applications for specific services of cable television or to applications distributed in the open market which are, therefore, outside the scope of this Recommendation.

5.3 Implementation of broadcasting TV, cable TV and IPTV specifications

The hybrid cable STB is required to support TV broadcasting, cable TV, and/or IPTV reception capabilities compliant with standards defined by each region. Examples of TV broadcasting specifications are described in Appendices V, VI and VII.

5.4 Service delivery architecture (informative)

Figure 5-3 illustrates the service delivery architecture assumed for the hybrid cable STB defined in this Recommendation. As Figure 5-3 indicates, the cable operator provides not only existing services, such as multi-channel video distribution, re-transmission of TV broadcasting, VoD etc., but also the hybrid services managed by the service delivery platform (SDP) and the next generation services where the hybrid cable STB is associated with mobile devices, such as smartphones and tablet PCs.

The STB defined in this Recommendation plays an important role in the provision of hybrid services in the above-described architecture, where presentation of various content and information provided by a cable operator, third parties and Internet, interactive services with users, etc., will be provided to the STB from the platform illustrated in Figure 5-3. In addition, synchronized services between the STB and in-home devices, such as NASs, PCs, tablets, smartphones, and DVRs connected via a home network, etc., Ethernet LAN or WiFi, are realized by interaction among devices. It is also possible for such devices having the same application environment as that of the STB to commonly use the same application in common.

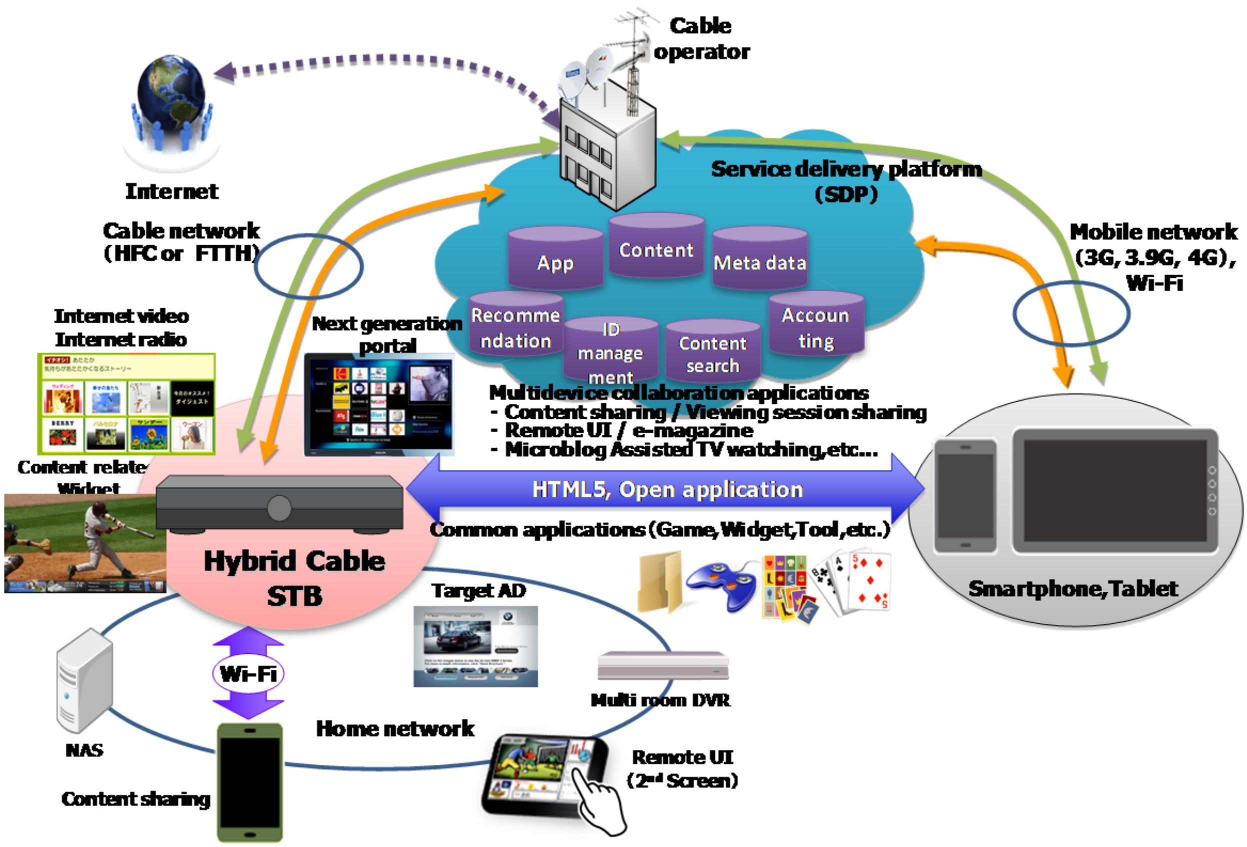


Figure 5-3 – Service delivery architecture

This description is intended to give an initial example of the service delivery architecture, and does not exclude other possible service provision schemes.

5.5 Environmental conditions

It is recommended that the hybrid cable STB be operated under the following environmental conditions:

– operating temperature range: 5 to 40 degrees Celsius

– operating relative humidity range: 10 to 85 percent (without condensation).

The following test conditions shall be referred to, in accordance with regional regulations.

[b-IEC 60068-1], [b-IEC 60068-2-1], [b-IEC 60068-2-2], [b-IEC 60068-2-6], [b-IEC 60068-2-14], [b-IEC 60068-2-27], [b-IEC 60068-2-30], [b-IEC 60068-2-31] and [b-IEC 60068-2-40].

6 Hardware

This clause defines the hardware for the implementation of the basic functions of the hybrid cable STB.

6.1 Tuner

The STB is required to have tuners for the reception of broadcast content and for a cable modem compliant with [ITU-T J.112], [ITU-T J.122], or [ITU-T J.222.0].

The feature "receiver for digital broadcasting" is described as follows:

– The tuner portion is required to support at least one of the broadcasting-related specifications defined in Appendices V, VI and VII, with an extension of the upper spectrum limit of 1GHz regardless of the standard selected.

– The STB can optionally support non-quadrature amplitude modulation (QAM) tuners in accordance with the broadcasting-related specifications defined by each region, as described in Appendices V, VI and VII.

The tuner output of the STB defined in this Recommendation is required to support simultaneous demodulation and decoding of at least two (2) broadcasting TSs at the same time. As an example of implementation, Table 6-1 describes the combination pattern of tuner usage when the STB is equipped with two QAM tuners (QAM 1, QAM 2) and two non-QAM tuners (non-QAM 1, non-QAM 2).

Table 6-1 – Combination pattern for simultaneous usage

Combination pattern	QAM 1	QAM 2	Non-QAM 1	Non-QAM 2
1	In use	In use	Not in use	Not in use
2	One of them in use		One of them in use	
3	Not in use	Not in use	In use	In use

6.2 Central processing unit (CPU) resource

The CPU resource is required to be capable of more than 2000 million instructions per second (MIPS) for execution of the application. The selection of CPU architecture is not included in this Recommendation.

6.3 GPU

The STB is required to implement the 3D graphic accelerator supporting OpenGL ES2.0

6.4 Memory

6.4.1 RAM

The STB is required to be equipped with more than 512MB RAM.

6.4.2 Flash ROM

The STB is required to be equipped with non-volatile memory greater than 1GB for the EPG, cache, firmware, applications, related storage of content, etc.

6.5 External interface

The hybrid cable STB is required to implement the following interfaces for external devices:

– Radio Corporation of America (RCA) pin

  The following items are required to be supported for analogue output:

  - NTSC/PAL/SECAM analogue composite video output (as defined in [ITU-R BT.1700])
  - Analogue audio output
  - A content protection mechanism for analogue video output (e.g., macrovision, CGMS-A, APS, etc.) is required to be compliant with the requirement defined by broadcasting regulations and/or guidelines defined by each region, as described in Annexes D, E, F and G.

– High-definition multimedia interface (HDMI)

The following items are required to be supported for HDMI output:

- HDMI Version 1.4a or above
- The authentication between devices should be EDID; signal encryption by HDCP. The control of linking devices conforms to CES (CEC 13.1)
- The STB is required to support a content protection mechanism for an HDMI output, such as high-bandwidth digital content protection (HDCP) compliance, with the requirement defined by broadcasting regulations and/or guidelines defined by each region, as described in Appendices V, VI and VII.

– Optical digital audio output

The following items are required to be supported for optical digital audio output:

- Digital audio interface as per [IEC 60958]
- Digital copy control
- [IEC 61937-4] in the case of transmission of MPEG-2 Audio Layer 2 [ISO/IEC 13818-3]
- [IEC 61937-6] in the case of transmission of advanced audio coding (AAC)
- The STB is required to support a content protection mechanism for digital audio output compliance, with the requirement defined by broadcasting regulations and/or guidelines defined by each region, as described in Appendices V, VI and VII.

– USB

The following items are required to be supported on a USB interface. USB usage and software are dealt with in clause 8.1.1.

- USB2.0 or higher
- USB host controller supporting the following host controller interfaces:

  EHCI (enhanced host controller I/F)

  OHCI (open host controller I/F)

  UHCI (universal host controller I/F).

– Ethernet

- The STB should be equipped with two ports accepting an 8pin modular jack (RJ-45) that supports the above 100 base-TX. It should also support the following combined uses:

  Combination 1: DOCSIS and LAN

  Combination 2: WAN and LAN

  Combination 3: LAN (Note: connected to WAN via LAN).
- The STB is required to support a content protection mechanism for LAN transport, such as DTCP-IP compliant with the requirement defined by broadcasting regulations and/or guidelines defined by each region, as described in Appendices V, VI and VII.

– Wi-Fi

The STB should be equipped for a Wi-Fi function with the same functionality as the above Ethernet LAN as an internal capability of the STB or via the external USB. It must support either access point (AP) mode or client mode (connected to the access point). The combination of enablement of the network interfaces can be configured only by the cable operator and the relevant customer support representative.

- Combination 1: Use Ethernet LAN but do not use Wi-Fi

- Combination 2: Use both Ethernet LAN and Wi-Fi (as an access point)
- Combination 3: Do not use Ethernet LAN but do use Wi-Fi (as an access point)
- Combination 4: Do not use Ethernet LAN but do use Wi-Fi (as a client)

As an access point, it is required to support both 2.4 GHz and 5 GHz [IEEE 802.11].

As a client, it is required to support all the following standards: 2.4 GHz and 5 GHz [IEEE 802.11], which will be automatically selected in accordance with the access point settings.

In addition to content transport with content protection via LAN, content transport over Wi-Fi is also permitted. In this case, the STB is required to support a content protection mechanism for Wi-Fi transport, such as DTCP-IP (the same as for LAN) compliant with the requirement defined by broadcasting regulations and/or guidelines defined by each region, as described in Appendices V, VI and VII.

6.6 Conditional access (CA) module interface

The STB is required to support a conditional access (CA) module interface compliant with broadcasting standards defined by each region, as described in Appendices V, VI and VII.

6.7 Remote control unit (RCU)

This clause specifies the remote control unit (RCU) to operate the STB. The RCU is required to be equipped with the buttons compliant with the requirements defined by broadcasting regulations and/or guidelines defined by each region, as described in Appendices V, VI and VII. However, in the case of a next generation RCU which is targeted to realize a further advanced user interface, vendors can implement it as their own product planning matter as long as there is no inconvenience to the user.

– Infrared (IR) type
  - RCU code is up to the vendor. RCU code should be provided to the operator.
  - A mobile phone like input method is required to be supported for the character input.
  - The RCU is required to reflect a hardware event such as a key press being reflected by an infrared (IR) LED blink.
  - The STB is required to recognize the IR RCU code 10 times per second.
  - The RCU is required to guarantee at least a 10 metre usable range.
  - The STB is required to detect IR signals from the RCU within 45 degrees horizontally and 15 degrees vertically.

– Radio frequency (RF) type
  - The STB is required to support the RF4CE interface for an RF type remote control unit (RCU).

– Home network type
  - The STB is required to support a mechanism to allow mobile devices, such as smartphones, to remotely control the STB through a LAN and/or Wi-Fi home network.
  - The STB is required to support the DLNA remote user interface (UI) function for the fundamental operations of a broadcast receiver, such as showing a portal screen, EPG control, channel zapping, playback of recorded programmes, trick play, etc.
  - In addition to the above-mentioned fundamental operations, the STB can optionally implement additional remote control mechanisms by means of application software. Details of such additional control mechanisms are outside the scope of this Recommendation.

  - The STB is recommended to support interactive operation between the STB and mobile devices, as well as the typical one-way operations described above. If supported, this capability is required to be implemented as application software. The details of this application software are implementation dependent, which is outside the scope of this Recommendation.

– Internet type

  - The STB is recommended to support a mechanism to allow mobile devices located outside the home, such as smartphones, to remotely control the STB through the Internet.
  - If supported, this functionality is required to be implemented as application software. The details of this application software are implementation dependent, which is outside the scope of this Recommendation.

6.8 Mechanical specification

Physically, the STB is required to be a space-saving design, easy to operate, and economical to produce.

– The STB is required to be equipped with the following physical switches or buttons:

  - Power On/Off
  - Channel Up and Down
  - Broadcast network selection, such as DTT, cable, IPTV, etc.
  - Hardware Reset

– The STB is required to be equipped with a CA card slot, as per clause 6.6.

– The STB is required to stand upright as well as lie down flat.

6.9 Mechanical robustness

The STB should be assembled with appropriate parts (such as screws) for the purpose of mechanical robustness and anti-tampering.

6.10 Feature of radio frequency (RF) input and (loop out) splitter

The feature of cable input (RF) should support clause 7.1 of this Recommendation. The following items should be supported for the condition/feature of RF distribution.

– Cable loop output: one or more

– Figure: Type F (75 ohm)

– Bandwidth: 75 MHz-1 GHz

– Distribution loss: 5 db or less

6.11 Power supply

The following items should be supported for the power supply of the STB.

– The power supply unit is recommended to be an external power adaptor.

– Power consumption should be selectable as either "Stand-by", "Wake-up", or "Normal'.

– The definition of each state should be in accordance with clause 16.

– The STB is required to be compliant with safety requirements defined by [ITU-T L.1000].

7 Software

7.1 External interface

The STB is required to install application software that can control devices through an external interface, as described in clause 6.5.

7.1.1 USB

7.1.1.1 Wi-Fi (USB wireless controller class)

It is recommended to install a wireless controller class driver to support USB Wi-Fi. See Table 7-1. The driver must support both the access point infrastructure mode (hereinafter referred to as AP mode) and the station infrastructure mode.

Table 7-1 – Linux WiFi driver example

Driver name	AP capability	Modes
ar9170	No	a/b/g/n
ath9k_htc	No	b/g/n
orinoco	No	b
p54usb	Yes	a/b/g
rt2500usb	Yes	a/b/g
rt2800usb	Yes	a/b/g/n
rt73usb	Yes	a/b/g
rt8187	No	b/g
zd1211rw	No	b/g

7.1.1.2 Other USB classes

The following classes shall be supported by the Linux kernel generic driver, or a compatible class driver shall be implemented:

- storage (USB mass storage class)
- camera (USB video class, input only)
- microphone (USB audio class, input only)
- hub (USB hub class).

7.1.2 HDMI

The STB is required to control an HDMI-connected device via HDMI CEC, see Table 7-2.

In order to control the STB from a connected device, an STB Power On/Off function (e.g., by an RCU or power button) is required. In order to control the connected device from the STB, the STB is required to send an On/Off CEC message. The API defined in clause 10 should allow sending of these messages.

Table 7-2 – HDMI CEC functions (Reference)

Function	Message
One Touch Play	Active Source/Image View On/Text View On
Routing Control	Active Source/Inactive Source/Request Active Source/Routing Change/Routing Information/Set Stream Path
System Standby	Standby
One Touch Record	Record Off/Record On/Record Status/Record TV Screen
Timer Programming	Clear Analogue Timer/Clear Digital Timer/Clear External Timer/Set Analogue Timer/Set Digital Timer/Set External Timer/Set Timer Program Title/Timer Cleared Status/Timer Status
System Information	CEC Version/Get CEC Version/Give Physical Address/Get Menu Language/Polling Message/Report Physical Address/Set Menu Language
Deck Control	Deck Control/Deck Status/Give Deck Status/Play
Tuner Control	Give Tuner Device Status/Select Analogue Service/Select Digital Service/Tuner Device Status/Tuner Step Decrement/Tuner Step Increment
Vendor Specific Commands	CEC Version/Device Vendor ID/Get CEC Version/Give Device Vendor ID/Vendor Command/Vendor Command With ID/Vendor Remote Button Down/Vendor Remote Button Up
OSD Display	Set OSD String
Device OSD Name Transfer	Give OSD Name/Set OSD Name
Device Menu Control	Menu Request/Menu Status/User Control Pressed/User Control Released
Remote Control Pass Through	User Control Pressed/User Control Released
Device Power Status	Give Device Power Status/Report Power Status
General Protocol	Feature Abort/Abort Message
System Audio Control	Give Audio Status/Give System Audio Mode Status/Report Audio Status/Set System Audio Mode/System Audio Mode Request/System Audio Mode Status/User Control Pressed/User Control Released
Audio Rate Control	Set Audio Rate

7.1.3 IP connection interface

7.1.3.1 WAN

When the STB uses a cable modem to connect it to the Internet via an HFC network, the STB should initialize it following [ITU-T J.112], [ITU-T J.122] or [ITU-T J.222.0] specifications (clause 7.5 Ethernet pattern (1) case).

When the STB uses an Ethernet terminal to connect it to the Internet via an FTTH network, the STB should configure the IP address, subnet mask, default gateway, and DNS server addresses following the specification below, (clause 7.5 Ethernet pattern (2) case):

– IPv4 : DHCP client [IETF RFC 2131]

– IPv6: Stateless automatic configuration [IETF RFC 4862] and DHCPv6 [IETF RFC 3315] for stateful automatic configuration.

In maintenance mode, either of the above two connections can be selected.

7.1.3.2 LAN

To connect the STB to a home network, it should be configurable for an IP address, subnet mask, default gateway, and DNS server addresses in either of the following methods, (clause 7.5 Ethernet pattern (1) and (2) case):

– IPv4 : DHCP client [IETF RFC 2131]

– IPv4 : Manual configuration

– IPv6 : Stateless automatic configuration [IETF RFC 4862].

In the case of clause 7.5 Ethernet pattern (3), the STB connects itself to the ISP provided Internet via a router on the home network, and the DNS server address should be able to be configured by DHCP [IETF RFC 2131], DHCPv6 [IETF RFC 3315] or manual setting.

Regardless of clause 7.5 describing the Ethernet connection pattern, the STB should work as a DHCP server [IETF RFC 2131]. In this case, the entire configuration is done manually and the initial value is dependent on the implementation.

7.1.3.3 WAN and LAN gateway

This clause is for the case of clause 7.5 Ethernet pattern (1) and (2).

This functionality can be turned on/off by the installer (operator) and by remote control.

The STB converts the IPv4 address and port number by NAPT to allow devices on the home network to access Internet through the STB LAN interface to the WAN interface.

The STB relays DNS queries from devices on the home network to the DNS server address which is acquired from WAN or LAN by DHCP.

7.1.4 Wi-Fi

The STB supports WEP, WPA and WPA2 encryption standards, and the following modes.

7.1.4.1 Wireless host: AP infrastructure mode

The STB works as an access point allowing the setting of SSID, ESSID and encryption keys (clause 7.5 Wi-Fi pattern (2) and (3).

When the embedded USB-attached WiFi interface supports both 2.4 GHz and 5 GHz bands (dual band), it should work as an access point simultaneously for both bands. The STB should support a Wi-Fi protected setup (WPS)-based Wi-Fi client automatic configuration, and the push-button and PIN code methods.

7.1.4.2 Wireless client

The STB is required to connect with the host as follows (clause 7.5 Wi Fi pattern (4)).

When the embedded USB-attached WiFi interface supports both 2.4 GHz and 5 GHz bands (dual band), it should list the entire SSID and ESSID acquired from both bands and use one of them for communicating through the access point. The STB is required to be able to:

– acquire all hosts' SSIDs and ESSIDs

– manually set the host SSID or ESSID; to choose an encryption standard (WEP, WPA, and WPA2) and to register the encryption key

– perform automatic configuration by either push-button or PIN code.

7.1.5 WAN-LAN connection function

To allow Internet connection to other LAN devices that belong to the same IP address segment as the STB LAN interface (wired/wireless) through the STB-embedded DOCSIS modem or WAN interface, the STB implements the following network service.

To adapt to various user-premises network environments, each service can be turned on/off in the SET setting menu.

- DHCP server : IP address assignment to LAN devices, and network information (such as DNS server) delivery (DHCP option).
- DNS relay: DNS query from LAN device, and response relay between WAN.
- NAPT: dynamic conversion of WAN side and LAN side IP address and port.

7.2 Management

The STB specified in this Recommendation is required to support SNMP-based management and the BBF specification described in clause 7.2.2.

7.2.1 Management by SNMP

The STB is required to support SNMP-based management mechanisms compliant with the cable television system specifications defined by each region, as described in Appendices V, VI and VII.

7.2.2 Management by BBF specification

7.2.2.1 Protocol and method

The STB specified in this specification applies [BBF TR-069] Version 1.2 for management protocol and method.

The STB specified in this Recommendation (CPE in [BBF TR-069]) applies [BBF TR-069] clause 3.1 for auto-configuration server discovery. The DHCP option 43 method described in clause 2 is the priority method. In this case, the DHCP server stores "dslforum.org" in option 60 and the STB needs to recognize this. In the event that it does not find ACS by DHCP, (including when it does not find the above string in option 60), a preset URL described in clause 1 is the second choice. In the event that it cannot be acquired (not set) the last choice is to use the default URL described in clause 3. (The default URL will be defined separately).

At the opening of a connection, it must support BBF [BBF TR-069] clause 3.2 methods; both begin from CPE and begin from ACS. For ID and passwords required in the ACS initiation case, digest authorization for ACS recognition will be defined separately.

To certify ACS by CPE, all of the following must be supported: basic, digest authorization and certificate (TLS). If TLS is used, CPE certification by a client certificate is also required in this Recommendation. (Operation will be specified separately).

In this Recommendation, after connection is established, message transport always uses [BBF TR-069] clause 3.3 describing TLS1.2. In other words, the URL schema given by ACS is always "https:"

A BBR document requires HTTP (with or without TLS) as the file transfer protocol. It must also support delivery by IP multicast for file transfer. File download from a broadcast must also be able to be performed by the specified scheme using this protocol.

For NAT, it is required to support [BBF TR-069] Annex G described method (STUN base).

Regarding the methods, the REQUIRED method that is described in [BBF TR-069] clause 3.6 of Table 5 "RPC message requirement", and the implementation of the following methods, are required in this Recommendation. See Table 7-3.

Table 7-3 – Method to be implemented to cope with [BBF TR-069] OPTIONAL RPC

CPE method	ACS method
Upload	RequestDownload
FactoryReset	Autonomous Transfer Complete (Option)
ChangeDUState (Option)	DUStateChangeComplete (Option)
	AutomonousDUStateChangeComplete (Option)

7.2.2.2 File transfer

The STB specified in this Recommendation is required to support the file transfer scheme presented in Table 7-4.

Table 7-4 – File transfer scheme

Item	Method	Initiative
Firmware upgrade Image	Download ScheduleDownload	ACS
Vendor Configuration File (Option)	Download (Option) ScheduleDownload (Option)	ACS
Vender Log File	Upload	ACS

7.2.2.3 Data model

In this Recommendation, BBS Device:1 Data Model Version 1.7 [BBF TR-157] Amendment 3, Data Model for a [BBF TR-069] Enabled STB Version 1.1 [BBF TR-135] Amendment 1, and [BBF TR-069] Data Model for Storage Service Enabled Device [BBF TR-140] Amendment 1, should be applied as [BBF TR-069] the data model. [BBF TR-140] Amendment 1 should be applied to a USB HDD or an internal HDD.

7.2.3 STB firmware management

The STB is required to support STB firmware downloading through a broadcast channel. The STB is also required to support [BBF TR-069] -based download control, where the actual downloading channel, i.e., via the IP network or via the broadcasting channel, will be specified by the URL scheme.

A background downloading capability is recommended (in this case, it is required not to cause any disturbance to the contents that are being watched or recorded).

It is recommended that firmware activation is to be done at the next boot by the user. If this requires a reboot, an appropriate message shall appear on the STB as a notification to the user, allowing the choice of reboot now or later.

The STB is required to support firmware download triggered by [BBF TR-069] as shown below:

– Download request from ACS

– Delayed download request from ACS

– Scheduled download request from ACS

The download protocol should follow the description of clause 8.2.2.

7.2.4 STB software module management

The STB specified in this Recommendation is required to follow [BBF TR-069] to manage installed and executed software modules.

Software package (Deployment Unit) status acquisition and management follows [BBF TR-157] Amd3 (CWMP Data Model Ver.1.7) Device.SoftwareModules.DeploymentUnit.{i}; Execution Unit status acquisition and management follows [BBF TR-157] Amd3 Device.SoftwareModules.ExecutionUnit.{i}.

Software package update (install, update, uninstall) uses [BBF TR-157] Amd3(SWMP Data Model Ver1.7) Appendix II "Software Module Management".

7.2.5 Other common requirement

The STB must enable an error log with the following reason code (RC):

- Irregular finish of application
- OS Kernel in Panic (option)
- Transmission error
- Error code defined by broadcasting related specifications of each region
- Hardware error (option).

7.3 Security

7.3.1 Scope of the protection

In this Recommendation, security protection is required for the following items:

- Content (including metadata): Data protection, copyright protection
- Software (operating system, middleware, application): data protection, execution authority protection, protection against unauthorized access
- Stored information (personal information, information relevant to the STB operation, log data): Data protection
- API: Security for the execution authority, protection against unauthorized access.

7.3.2 Assumed security risks and solutions

Table 7-5 shows assumed security risks and solutions. The STB shall implement the necessary functionality to realize these security solutions.

Table 7-5 – Security risks and solutions

Covered area		Risks	Solutions
Content	STB storage area	Piracy, unauthorized copy, content tampering	Encryption (tamper resistant)
	Composite output		CGMS-A, macrovision
	HDMI output		HDCP
	LAN		DTCP-IP
	WAN	Hacking	Filtering, authentication
		Content interception on the transmission channel	CAS (clause 6.6)
	USB	Piracy, unauthorized copy, content tampering	Content encryption by STB
		Playback with unpermitted terminals	Local encryption

Table 7-5 – Security risks and solutions

Covered area		Risks	Solutions
Software	STB	Content tampering	Depends on implementation
		Unauthorized promotion of Access privilege	
		Internal information interception	
		Overloading attack	
	LAN	DoS attack	
		STB hijack	
	WAN	DoS attack	
		STB hijack	
	USB	Internal information interception	
		Illegal data input from outside	
Stored information	STB storage area	Tapping, data tampering	STB identification (clause 7.3.3) Encryption of BBF [BBF TR-069] remote management (clause 7.2.2.1)
	LAN	Tapping, data tampering on LAN network	
	WAN	Tapping, data tampering on WAN network	

7.3.2.1 Risks and solutions related to API

Regarding the security risks related to APIs, proper management is necessary based on the operation policy and management rules of the cable operator. Assumed risks are shown in Table 7-6.

Table 7-6 – Security risks and solutions regarding API

Covered area		Risks	Solutions
API	STB	Illegal access	Depends on the implementation
		Data tampering	
	LAN	Tapping, data tampering on LAN network	
	WAN	Tapping, data tampering on WAN network	
	USB	Leak of internal API information	
		Illegal API access from outside	

7.3.3 Receiver identification

The STB defined in this Recommendation shall hold the receiver ID (6 bytes). This ID shall be used as the password used by the illegal usage prevention function and the authentication in [BBF TR-069], and shall be unreferrable by an external party. The user-ID for [BBF TR-069] shall be <OUI><ProductClass><SerialNumber> as described in [BBF TR-069] clause 3.4.4.

7.3.4 Illegal usage prevention function

The STB defined in this Recommendation shall accommodate an illegal-use-prevention function based on EMM. This STB shall properly use either one of two (or more) QAM tuners to acquire the illegal-use- prevention control EMM periodically delivered on the broadcasting signal. Apart from the reception of broadcasted content, this illegal usage prevention function is applied for the DLNA function and for application SW startup, as well.

7.3.5 User identification

If the user accesses the STB defined in this Recommendation from such facilities as a cell phone or a pad terminal, those terminals should identify the user by ID/password, and that information should be communicated to the STB. During the operating session, the STB should keep that information.

7.3.6 Service and content protection (SCP)

7.3.6.2 Service protection

The STB defined in this Recommendation is required to support service protection mechanisms compliant with broadcasting-related specifications defined in each region, as described in Appendices V, VI and VII. The STB is required to support [ITU-T X.1193] for a key management scheme for secure IPTV services. As for the secure transcodable scheme of IPTV, the STB is required to support [ITU-T X.1192]. The STB is also required to support [ITU-T X.1195] for interoperability between multiple SCP mechanisms.

7.3.6.2 Content protection

The STB defined in this Recommendation is required to follow the rules for copy control and content protection that are compliant with the broadcasting-related specifications defined in each region, as described in Appendices V, VI and VII. The STB is required to support [ITU-T X.1191] for content protection of IPTV services.

7.4 Home network

The STB shall support content sharing functionality over the home network based on [IEC 62481-1], [IEC 62481-2] and [IEC 62546]. In this clause, specific technical requirements that shall be adapted to the STB are described.

7.4.1 DLNA device class

The STB is recommended to support DLNA device class service.

7.4.2 DLNA media profile

For each device class, the STB shall support media profiles corresponding to each region. Required profiles are:

– TS format defined in AV Class: MPEG-2 Profile of [IEC 62481-2] (such as MPEG_TS_JP_T, MPEG_TS_HD_NA_XAC3, MPEG_TS_HD_KO_XAC3, MPEG_TS_SD_EU_AC3_T, etc.)

– TS format defined in AV Class: MPEG-4 Part10 (AVC) Profile of [IEC62481-2] (such as AVC_TS_JP_AAC_T, AVC_TS_HP_HD_AC3_T, AVC_TS_SD_EU_T, etc.).

Also, in the above profiles, if service_type is defined as the digital radio sound service, and if the DMS does not support that stream, the STB shall not send it to the DMS.

If the STB supports additional profiles other than the above, the STB can check the profile availability of DMS in many ways, (e.g., the STB requests CDS:X_GetDLNAUploadProfiles to the DMS), but this depends on the implementation.

The STB shall implement the DMS function as follows:

– When the STB stores content uploaded from an external device, the STB shall store it in the media format defined in the AV Class MPEG-2 Profiles of [IEC 62481-2].

– When the STB stores broadcast contents, it shall support at least one of the following methods:

  - Direct recording as MPEG-2 media defined in AV Class: MPEG-2 Profiles of [IEC 62481-2].
  - Transcoding to the AVC format defined in AV Class: MPEG-4 Part10 (AVC) Profile of [IEC 62481-2].

However, when the STB runs video transcoding in the latter situation, it shall proceed with special consideration for resource conflicts and error handling.

7.4.3 Connection number and avoidance of resource conflict

To achieve the proper control of DLNA connections and to avoid resource conflicts, the STB shall comply with the following technical requirements:

– The STB shall support simultaneous transmission of multiple connections. It shall have the capability for at least one upload connection (from STB to external device) and one download connection (from external device to STB) simultaneously. It may have the ability to support more connection channels. In such a case, the STB shall control its connection number as follows:

  - The STB shall give higher priority to the upload recording than to the transmission of general content to DMP/DMR. Therefore, when upload recording is carried out and the connection number exceeds the connection upper limit of the STB, the STB shall cease general content transmission and shall execute upload recording instead.
  - When all decoders/transcoders are busy due to the simultaneous recoding, and the STB does not have any remaining capability for content decoding, it shall display an error message indicating that there is no more DMP/DMR capability.
  - The connections for the content list acquisition and remote UI are not included in the above limitation.

7.5 3D viewing

HDMI video output shall be set to 3D display mode and the content shall be played back using a 3D-based display in the screen region assigned to 3D visual contents in either of the two following cases:

1) playing back video, including linear TV and contents delivery via an IP network, when the 3D identification signal (described in clause 9.4) is included in the video signal;

2) when an application with 3D visual output (3D application) is started up.

The visual output based on HDMI shall be changed to 2D display mode at the time when the playback of 3D visual content is finished or stopped, or the 3D application terminates.

Under 3D mode, presentation of other types of contents (except for 3D video), such as graphics and user interfaces, should support one of the following display modes. To avoid any oddities in the 3D display, these will be overlayed on 3D video following the 3D display mode. For example, when in side-by-side mode, the graphics/user interface must be presented in the same mode.

If software (SW) or hardware (HW) has any restrictions during the playback of 3D video, it should support one of the following:

- showing of only 3D video (no other contents are shown). At the time of finishing or stopping the playback of 3D video, the mode shall return to 2D mode.
- displaying of 3D video as 2D video, with other contents.

7.6 Audience measurement

In order to measure the user's activity information (TV watching activity, etc.) over the STB by the cable operator, it is required to apply the audience measurement functions described in this clause. This clause describes the technical method for audience measurement.

7.6.1 Architecture

As described in clause 8 of [ITU-T H.741.0], the architecture of audience measurement is found in Figure 7-1.

Also, as defined in [ITU-T H.741.0], the abbreviations used for the audience measurement functions are:

– Terminal Device-Audience Measurement Function (TD-AMF)

– Home Gateway-Audience Measurement Function (HG-AMF)

– Network Function-Audience Measurement Function (NF-AMF)

– Service Control-Audience Measurement Function (SC-AMF)

– Content Delivery-Audience Measurement Function (CD-AMF)

In this Recommendation, as shown in Figure 7-1, the hybrid cable STB acts as the end-user functions instead of the IPTV terminal functions. The STB is equipped with the TD-AMF, and the ACS of [BBF TR-069] provides aggregation functions. In this Recommendation, the TD-AMF for the hybrid cable STB, and the interface between the TD-AMF and the aggregation function, are defined. Other interfaces and audience measurement functions are beyond the scope of this Recommendation.

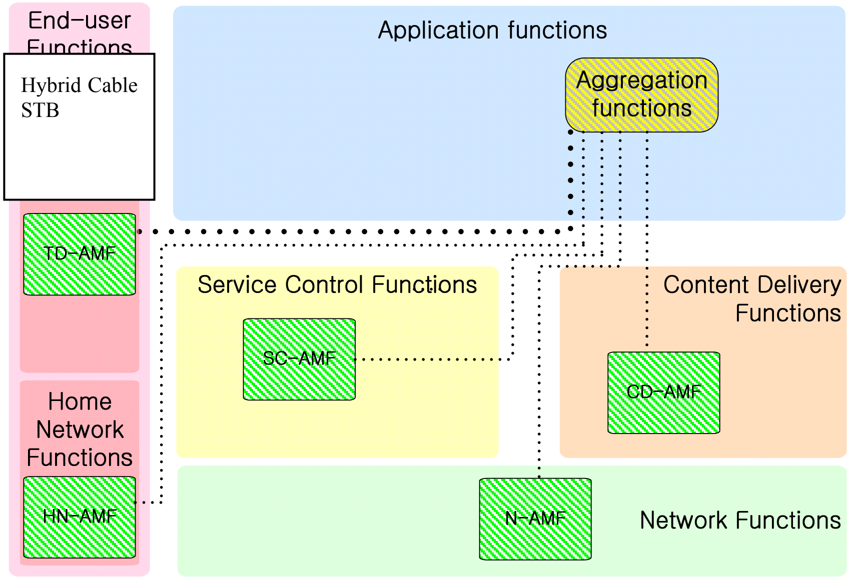


Figure 7-1 – The relationship between the hybrid cable STB and the audience measurement architecture defined in [ITU-T H.741.0]

7.6.2 Audience measurement items for the STB

Table 7-7 shows the assumed audience measurement items that will be measured from the STB.

Table 7-7 – Overview of assumed audience measurement items

Items	Description	M/O
TerminalDevice ID	Inform MAC address of STB as "TerminalDeviceID" element.	M
Hardware information	Inform hardware information as "DeviceInformation" element.	M
STB information	Inform product name of STB, STB specific ID, etc.	M
TV information	Inform TV information (such as manufacture ID, product number, etc.). STB gets these information through HDMI interface.	O
Channel change event	Inform events related to the linear TV watching as "ChannelStartEvent" and "ChannelStopEvent" element.	M
Power on/off event of Display	Inform the display (TV) power on/off event acquired from HDMI interface as "DisplayStatus" element.	M

Regarding the above measurement items, [b-ITU-T H.741.2] and [b-ITU-T H.741.3] contain more information, including the message format. In addition to those listed in Table 7-1, some other measurement items, (e.g., recording history, time shifted content watching, trigger information of each event, etc.), may be measured and reported. Supplemental information for the implementation, especially concerning the message format of these additional items, is described in Appendix I.

7.6.3 Delivery protocol of audience measurement data

As described in clause 7.2.2.1, the hybrid cable STB is equipped with the [BBF TR-069] function, and [BBF TR-069] includes the file upload capability. Therefore, in this Recommendation, it is required to use [BBF TR-069] as the delivery protocol for audience measurement messages. The STB corresponds to the CPE, and the server, which has an aggregation function, corresponds to the auto configuration server (ACS) of [BBF TR-069]. It is recommended that the configuration message be transferred through the "Download" method of [BBF TR-069] and that the measurement report message be transferred with the "Upload" method of [BBF TR-069]. Details are described in Annex A

7.6.4 Delivery policy

All communications (including session establishment, retry policy, etc.) between the STB and the ACS are required to comply with the [BBF TR-069]. In addition, the STB is required to possess a function to prevent bursty access to the ACS.

7.6.5 User permission for the audience measurement

To gather audience measurement data from the STB, the user's permission is required. The levels of user permission are defined in [ITU-T H.741.0] as follows:

– Permission level 0 – No end-user information.

– Permission level 1 – Only distinguishable from other end-users.

– Permission level 2 – Distinguishable and non-identifying generic end-user information.

– Permission level 3 – Identifying user information and non-identifying generic end-user information.

Where:

– Distinguishability is supported by the use of anonymous user identifiers.

– Non-identifying generic end-user information is extensively supported without specifying the information values.

– Identifying end-user information may be the identifying information about the actual end-user or subscriber. It may be easily combined to identify the end user. End-user information is classified either as personal data which requires control or that which does not. Identifying end-user information requires control and is fully specified and marked as "controlled" information.

7.6.5.1 Transmission method of user permission

The STB is required to obtain the proper user permission corresponding to the data that will be measured. However, informing the users of permission, as in the user permission message defined in [ITU-T H.741.0], is not required in this Recommendation. The user permission parameter shall be included in the measurement report.

7.6.5.2 The behaviour of unpermitted audience measurement

If the STB does not obtain proper user permission for the required measurement, the STB is prohibited from sending any private information. In this case, when the STB receives any request for a measurement report, the STB is required to send a measurement report message that contains attributes for only the terminal device ID and permission level.

7.6.6 Handling of measurement data

The following instructions are to be satisfied for retaining audience measurement data in the STB.

– Measurement data cannot be taken out from the STB except by the measurement process described above.

– Erasable data (specified by contract between the cable operator and the subscriber) can be erased only by the subscriber.

– All the retained measurement data can be erased in the maintenance mode of the STB and by remote control by the cable operator.

7.7 Software reset

To deal with STB freezing or software malfunctions, it is required that STB operation can be recovered by software reset (re-boot). Even in a failure condition in which applications cannot use bi-lateral communications, such as unavailability of WAN communication (used for DOCSIS cable modem and Ethernet terminal), it is required to be able to receive programmes, to record, and to make reservations for viewing.

7.8 DVR and related function

Regardless of the internal accommodation of a physical HDD, it will be possible to use the following functions:

– Recording functionality (the choice of the recording device and implementation are implementation dependent, and outside the scope of this Recommendation).

  • For recording to an internal HDD or external HDD with a USB connection, the STB is required to support a local encryption mechanism compliant with the broadcasting standard defined by each region, as described in Appendices V, VI and VII.

  - For the recording to a removable media, the STB is required to apply a content protection mechanism for the corresponding media that is compliant with the broadcasting standard defined by each region, as described in Appendices V, VI and VII.

– Recording in parallel with the recording of another programme or playback
  - Recording of one programme in parallel with watching another programme
  - Recording of two programmes in parallel with watching one of them.

– Recording reservation and reminder reservation
  - "Programme reservation" specifying date&time or programme name
  - "Event reservation" specifying programme to be recorded on EPG
  - Confirmation and deletion of reservation(s) on screen
  - Automatic update of scheduled recording and reminder by programme start/end time update
  - Notification on screen to watching (STB-operating) user shall be given on the screen before starting dual recordings
  - Lock of remote control keys while executing scheduled recording/reminder (except in the case of cancellation, the user should be notified on screen that operation is being prohibited).

– Parental control functions (including access from external devices (refer to clause 8.4))
  - Restricted list viewing function available only to specific users
  - Viewing history of a user's controlling function
  - Viewing age control restriction as part of parental control

  In addition, it is recommended to implement the following implementation:
  - Compression of recorded contents
  - Real-time transcoding to lower bit rate at recording
  - Transcoding to lower bit rate after recording.

Concerning the actual compression methods, the specification of the description of the bit rate parameter is beyond the scope of this Recommendation. However, compression methods shall be compatible with DLNA described in clause 7.4.

7.8.1 Concurrent operation and congestion avoidance

The STB defined in this Recommendation is required to support the concurrent operation of background processes such as television programme recording, bit rate conversion (transcoding), scheduled recording control from remote, etc. In addition, it is also required to guarantee that such background processes do not put any influences onto foreground applications or functionalities by causing loss of smoothness, slowdown, or other noticeable impairment.

When an electronic portable device with display capability, such as a mobile phone or a tablet, is used as a remote controller of the STB, it is required to support the following two types of operations:

– Type 1: User interaction, such as operation and getting response, can be performed only on the portable device, with nothing changed or displayed on the STB screen.

– Type 2: User operations can be performed on the portable device, and their results will be shown on the STB screen (and also on the portable device if necessary).

The STB defined in this Recommendation is recommended to implement the following behaviours to ensure schedule recording and play-back operations by avoiding possible congestions:

– If the number of simultaneous recordings, or the number of simultaneous recordings and viewing, is greater than the number of tuners in the STB, it is required to present an error message to inform the user that not all of the operations can be successfully carried out. In this case, the way to manage this congestive situation, for example by cancelling some previously scheduled recoding or viewing at the user's option, or not allowing any additional operations related to scheduled recoding or viewing, is dependent on product planning or implementation.

– It is recommended that the STB defined in this Recommendation manage hardware resources such as CPU, memory, hard disk drive I/O bandwidth (regardless of internal or external HDD), etc., in such a manner as to prioritize the recording or viewing functionality.

7.8.2 Transcoding

The STB defined in this Recommendation is required to support the following capabilities that will be carried out simultaneously with recording, or subsequently after recording, with an embedded transcoder implemented as hardware or software:

– bit rate conversion for long-time recording

– resolution and/or frame rate conversion for playback on a mobile device such as a smartphone or a tablet. It is required to store the converted programmes in a format conforming to the DLNA media profile defined in clause 7.4.2.

The following defines requirements for the transcoder of the STB for long-time recording:

– It is recommended to provide the same number of transcoders for long-time recording as there are tuners.

– If the number of transcoders is smaller than the number of tuners, it is required to perform background transcoding of a recorded programme that was not transcoded in real time. In this case, there may exist both the original stream and the converted stream that may reside in storage. Therefore, a disk space availability check for this purpose is required at the time of setting a recording schedule.

– In the above-described case, it is recommended to support playback or DLNA remote playback of the original stream, even when the stream is in the background transcoding process, to provide (as far as possible) the congestion avoidance described in clause 7.4.3.

The following defines requirements for the transcoder of the STB for portable device playback:

– It is recommended to support at least one real-time transcoding process of the programme for portable device playback.

– If the number of the transcoders for portable device playback is smaller than the number of tuners, it is required to perform background transcoding of a recorded programme that was not transcoded in real time. If the programme is specified to be transcoded for long-time recording, as well as for portable device playback, the congestion avoidance is required in accordance with the description below.

– It is also recommended to support playback or DLNA remote playback of the original stream, even when the stream is in the background transcoding process, in order to provide congestion avoidance described in clause 7.4.3.

– Either the original stream or transcoded programme for long-time recording can be transcoded into a converted stream for portable device playback. If the programme is specified to be transcoded for long-time recording, as well as for portable device playback, the congestion avoidance is required in accordance with the description below.

It is required to manage the scheduling of background transcoding so that it does not affect another fundamental functionality, such as recording. For example, background transcoding will be carried out only when no other recording processes are running or are in the standby state.

If a programme is specified to be transcoded for long-time recording as well as for portable device playback, and only one transcoding process is available, the congestion avoidance is required in accordance with the description below.

– Product planning or implementation will determine which transcoding process will perform in real time or in the background. It should be noted that a certain amount of HDD free space will be necessary if the original stream needs to be temporarily stored in the real-time transcoding process.

7.9 Basic function of system software

7.9.1 File system

This Recommendation does not specify the file system utilized in the STB. The file system available for initiating the STB will be selected and set according to the OS.

Any external storage connected via USB (conforming to the USB Mass Storage Class) shall be capable, at least, of reading and writing partitions of FAT16 and FAT32, in order to remain compatible with such general digital devices as digital camera and USB HDDs. However, this condition may not be applied to the storages to be used as the recording medium for the DVR function (refer to clause 8.8).

7.9.2 Working log

Regarding the STB internal operation, the following items shall be recorded in the logs. The information specified by these item is referred to for the purpose of software management, following the Broadband Forum [BBF TR-069]/[BBF TR-106] specification (refer to clause 8.2.5).

– History of system startup and error(s)

– History of download of system update, adaption and error(s)

– History of NW connection (DOCSIS, WAN, LAN) and error(s)

– History of functional activity relating to WAN-LAN interconnection and error(s)

– History of installation of application(s), invocation and error(s)

– History of functional operation of broadcasting function and error(s).

7.10 STB setting

The STB shall comply with the following setting methods:

– Channel setting (refer to clause 7.1)

– Cable modem setting (refer to only ON/OFF, refer to clause 8.1.5)

– NW setting (refer to clauses 8.1.5, 8.1.6 and 8.1.7)

– Initial access destination for each service ([BBF TR-069] server, IPTV initial server, Audience measurement server etc., in case the autonomous acquisition is not possible)

– Setting and deletion of user information (zip-code, age, PIN code)

– Other user individual information, (user ID allocated by the operators; detailed individual information, including name, address, date of birth, sex, mobile phone number etc.; information related to the account settlement with credit card number; preference information, including hobby, preferable programme genre, etc.), shall be managed by the operators or their relevant platform and should not be kept in the STB.

– User information, (prior registration of fingerprints; facial picture; voiceprint, etc.), relevant to the next generation user authentication technology (if accommodated).

The STB shall be able to obtain the internal information as follows:

– Firmware (OS, framework) version

– List of installed application SW and version of each.

7.11 Maintenance mode

The hybrid cable set-top box is required to set the following items.

– Deletion of personal data

– Firmware up-date by USB memory

– Setting of cable modem (ON/OFF only)

– Setting of initial access for services (BBF [BBF TR-069] server, URL of portal site, IPTV initial server, audience measurement server, etc., in the case of non-automatic operation)

– Operator specific setting and customization of portal screen.

8 Media handling

8.1 Decoder (for broadcasting type service)

The STB defined in this Recommendation is required to support a media decoder for broadcasting type services defined by broadcasting-related specifications of each region, as described in Appendices V, VI and VII.

8.2 Transcoder (for broadcasting type service)

The STB defined in this Recommendation is required to perform real-time execution of transcoding, including the following items:

– If the format of interlace/progressive is different in source/destination streams, i/p or p/i conversion should be performed properly. Comb-like noise for horizontal movement or a judder problem for vertical movement must be avoided.

– If the resolution is different in source/destination streams, the aspect ratio of codec resolution must be the same after transcoding, i.e., after the transcoding, the aspect ratio of a pixel is always 1:1. For example, when transcoding a signal of codec resolution 1440 × 1080, with a display aspect ratio of 16:9 to codec resolution 640 horizontally, the vertical codec resolution should be 360. Of course, the aspect ratio indicator should be a specific value that indicates 16:9.

– The encoding format after the transcoding must support the following.

  • Resolution, frame rate and scan method (i/p) is the same before and after transcoding

  • FWVGA(854 pixels × 480 lines), frame rate 30 pcs/sec, scan type progressive, (bit rate is up to vendor)

  • Other format after transcoding (coding method, coding resolution, frame rate etc.) are out of the scope of this Recommendation. If multiple conversion formats are available, the user should be able to choose one of them.

8.3 Decoder (for an IP interactive service and applications)

In addition to the transcoder defined in clause 8.2, the STB is required to support the combinations of video decoder, audio decoder, and multiple format described in Tables 8-1, 8-2 and 8-3, which have been assumed to be utilized in an IP interactive service or by applications.

Table 8-1 – Example of decoder video format

Multiplex format	Video decoder	Audio decoder
[ISO/IEC 14496-12] (ISO base) /[14496-14](.mp4) /[14496-15](avc file)	[ITU-T H.263], [ITU-T H.264] | [ISO/IEC 14496-10] HP@L4/SHP@L4.1 [ISO/IEC 14496-2] (MPEG-4 Visual) SP	[ISO/IEC 14496-3] (MPEG-4 AAC), HE-AAC v1/v2

3GPP(.3gp)
3GPP2(.3g2)
	Same as above (except for SHP@L4.1)	In addition to above, AMR-NB,AMR-WB

Flash Video(.flv)	Sorenson Spark, On2 VP6, [ITU-T H.264] | [ISO/IEC 14496-10] HP@L4	MPEG-1/-2 Audio Layer3 (MP3), [ISO/IEC 14496-3] (MPEG-4 AAC), HE-AAC v1/v2
WebM(.webm)	On2 VP8	Ogg Vorbis

Table 8-2 – Example of still image format

Support format
[ISO/IEC 10918-1] (JPEG-.jpg) baseline/progressive
GIF(.gif)
W3C Recommendation Portable Network Graphics(PNG-.png)
Microsoft BMP (.bmp)
WebP(On2 VP8 + AV1)
CIPA DC-007 Multi-Picture Format(.mpo)

Table 8-3 – Example of audio format

Multiple format	Audio codec
[ISO/IEC 14496-12] /[14496-14](.mp4,.mp4a), 3GPP(.3gp),3GPP2(.3g2)	[ISO/IEC 14496-3] (MPEG-4 AAC), HE-AAC v1/v2, AMR-NB,AMR-WB
MP3(.mp3)	MPEG-1/-2 Audio Layer 3 (MP3)
Microsoft WAV(.wav)	Linear PCM (8 bit/16 bit, mono/stereo)
Ogg(.ogg)	Ogg Vorbis
AIFF-C(.aifc)	PCM (16 bit)

8.4 3D content identification

The STB is required to support the 3D content identification methods described in Table 8-4.

Table 8-4 – Identification method of 3D content

3D Picture format	Compress method	3D identification	Reference
Side-by-Side	MPEG-2		
	[ITU-T H.264]	Frame Packing Arrangement SEI	[ITU-T H.264] Annex D (D.1.25)
Full HD (temporal interleaving frame arrangement)	[ITU-T H.264] MVC	Frame Packing Arrangement SEI	[ITU-T H.264] Annex H

3D signalling that delivers L and R frames alternatively is called "Frame Sequential", but in [ITU-T H.264] it is described as "a temporal interleaving frame arrangement" and thus this Recommendation uses this term.

8.5 Codec (for the function of video phone)

Following the functional requirement document, the STB is recommended to implement video phone functionality. If this is implemented, the codec (encoder and decoder) that is specified by the video phone application is required to work in real-time on the STB.

Some commercialized videophones (non-PC devices) adopt mainly [ITU-T H.264] | [ISO/IEC 14496-10] (MPEG-4 AVC) for video codec and SILK (IETF draft-vos-silk-00.txt.) for audio codec.

9 Application environment

9.1 Overall architecture of the application environment

The application environment for the STB defined in this Recommendation consists of a virtual machine-based application platform for procedural applications and a browser-based application platform for declarative applications.

It is recommended that an STB equipped with a chipset having hardware acceleration capabilities in terms of the following functions, should allow applications to take advantage of such acceleration capabilities through their APIs to provide high performance and advanced visual presentations to users:

– Rendering of graphics and user interface (e.g., 2D and 3D rendering by OpenGL ES)

– Encoding and decoding of video and audio.

It is recommended that the STB implement the latest version of modules for the application environment, such as HTML 5, CSS, JavaScript, Java, etc., at the time of shipping of the STB. It is also recommended that the STB has the capability to update relevant modules.

9.2 Legacy support

Only in response to the operator's specific demand can the existing application environments that are not defined in this Recommendation be optionally implemented in the STB for the sake of the legacy application support.

It should be noted that the following situations may arise when the legacy application support functionality is implemented in the STB:

– Concurrent operation of legacy applications, and the applications running on the application environment, defined in clauses 10.3 and 10.4 of this Recommendation, may not be possible.

- Functional enhancement of legacy applications with the functionalities, defined in clauses 10.3 and 10.4, may not be possible.
- Seamless switching between legacy applications and the applications running on the application environment, defined in clauses 10.3 and 10.4, may not be guaranteed.

Furthermore, application environments defined by [ITU-T J.201] and/or [ITU-T J.202] are required to support the handling embedded applications transferred in-band with broadcasting RF signals or portal screen technologies already deployed in the existing broadcasting systems. When any such application environment is activated, the STB defined in this Recommendation can only show such applications, and is not required to simultaneously present those applications with other applications running on the environments defined in clauses 10.3 and 10.4, or legacy applications.

9.3 Declarative application environment

It is recommended that the STB defined in this Recommendation support the Open IPTV Forum Release 2-Volume 5: Declarative Application [b-OIPF DAE] as a declarative application environment.

It is recommended that the application environment be built upon the latest web technologies, such as HTML 5 and JavaScript. It is also recommended that the application environment is implemented in an upgradable manner to keep up with the future standardization activities of the Web on TV in W3C, such as HTML 5 plus extended functionalities for TV-related applications.

9.4 Procedural application environment

9.4.1 Class requirements

It is recommended that the STB defined in this Recommendation support the virtual machine-based Java applications. The following frameworks are required to be supported:

- Activity Manager
- Window Manager
- Content Providers
- View System
- Package Manager
- Resource Manager
- Notification Manager
- Telephony Manager
- Location Manager

The frameworks related to such functions, which are not actually implemented in the STB, are not necessarily required to perform as they are intended. However, these frameworks are required to be implemented with dummy behaviours to avoid unexpected application crashes, even when an application that calls non-implemented functions is executed on the STB. Some Java packages related to telephony-CDMA and GSM are not required for the STB services.

9.4.2 Library requirements

This Recommendation defines the library as a middleware consisting of binary form CPU instructions placed in the native layer, which is a lower layer than the virtual machine layer. The following defines types of libraries and library requirements:

Standard library: A library equipped with APIs for native access to provide fundamental functionalities necessary to execute standard applications.

Extended library: A library to provide STB specific functionalities such as television signal reception, home networking, remote management, etc. APIs for these extended functionalities are defined in clause 9.4.3 and will enable applications that will be able to take advantage of these extended functionalities. It is assumed that disclosure of these extended APIs to the public, and/or access permissions to these APIs, can be managed by cable television operators.

9.4.3 Extended APIs

Table 9-1 defines the APIs required for extended functions specific to the STB.

Table 9-1 – Extended APIs

Category	Description
Channel information	Acquisition and update of channel information, including scanning of broadcasting signals.
	Seeking of channel information.
Channel selection	TV channel selection
	Acquisition of information related to TV programme shown on the screen.
EPG	Acquisition and update of EPG
	TV programme search by keyword(s), e.g., title, genre, cast name, etc.
Schedule Recording and Viewing	Schedule recording and its transcoder settings Schedule viewing
	Acquisition and modification of a list of schedule recording or viewing
	Immediate recording and halting
	Playback of recorded programmes
Home networking (DLNA)	DMP: Acquisition of a content list from DMS
	DMP: Browsing of a list of content stored in DMS
	DMP/DMR: Playing back of content stored in DMS
	DMP: Control of upload recording
	DMR :Playback control initiated by a request from DMC
	DMS :Access permission settings of DMS
	DMS: Handling of content browse request from DMP or DMC
STB-related information	Acquisition of CAS-ID
	Acquisition of MAC address
STB settings	Setting of portal URL
	Privacy settings, particularly for the audience measurement functionality
	Parental control settings
	Factory reset
Remote management	Setting of ACS URL
	Invoking of ACS methods
	Handling of CPE methods invoked from remote ACS

9.5 Transport of applications

The STB defined in this Recommendation is required to support two types of application transport schemes as follows:

1. The STB downloads an application through an out-of-band transmission channel, such as DOCSIS IP transport, where such downloading is initiated by an event message [ITU-T J.201] transmitted within in-band with respect to broadcasting signals.
   - The primary purpose of this scheme is to achieve application behaviour control synchronized with television programmes currently being received and presented.
   - To achieve this purpose, it is required to implement application control mechanisms where application downloading, launching, and halting can be controlled in accordance with the received event message.
2. The STB downloads an application through an out-of-band transmission channel, such as DOCSIS IP transport, where software module management and downloading initiation is controlled by [BBF TR-069] ACS.
   - The primary purpose of this scheme is to allow cable television operators to add or remove applications from the head-end side. This scheme can be also used for application management such as updating and maintenance by operators or application vendors.

10 Applications for basic functions

10.1 Electronic programme guide (EPG)

The EPG function is required to support the aforementioned broadcast services without the user recognizing the network ID. The following functions shall be provided.

10.1.1 Display of EPG

A grid display is required for the programme list from the current programme to the scheduled programme. The current programme shall be emphasized by re-sizing or altered colour.

10.1.2 Programme viewing

The STB is required to enable the user to select display of details of the programme guide to be displayed, or the programme itself, when the user selects the currently broadcast programme.

10.1.3 Programme search

The STB is required to search the title of the programme by keyword or by broadcast genre, as described in [b-ARIB STD-B10] (content_nibble_level).

10.1.4 Recording reservation from EPG

A recording reservation shall be made by one click from the EPG. A recording setting function is required.

10.1.5 Restricted viewing

Displaying restricted content shall be controlled in accordance with the parental rating control of the STB.

10.1.6 Interaction with VoD, DVR, or browser

The STB is required to provide the following optional services by applications:

- Interaction between EPG and VOD
- Interaction between EPG and DVR function

– Interaction between EPG and HTML browser
– Recommendation function
– Interaction with mobile tool.

10.2 Recording function

The recording function is required to record or to reserve a broadcast programme in the recorder. The following functions shall be provided:

– Instant recording: instant recording of current broadcast programme
– Programme reserved recording: recording by event schedule
– Time reserved recording: required assignment of day, time and channel
– Series recording: automatic repetition of recording reservation following given condition
– Setting of recording: automatic recording function under given condition.

10.2.1 Basic recording function

Following functions are required to STB:

– Terrestrial broadcast and satellite broadcast programme can be recorded
– SD and HD pictures can be recorded. HDD direct recording of digital broadcast bit stream is also required
– Transcoding of SD/HD pictures is required
– Supporting function of commercial recorder and NAS
– Recording reservation using EPG
– Simultaneous processing of programme viewing, replaying and recording.

The required level for simultaneous processing of programme viewing, replaying and recording is shown in Table 10-1.

Table 10-1 – Simultaneous processing

	Back ground			
	Viewing	**Replaying**	**Recording**	**External output**
Viewing	R	R	M	R
Re-playing	R	R	M	R
Recording	R	R	R	R
Time shift	R	R	R	R

Recording related functions are shown in Table 10-2.

Table 10-2 – Recording related functions

Function	**Required level**	**Explanation**
Keyword search	M	Programme search and recording reservation from EPG can be done by keyword
Programme navigation (Video)	O	Video can be used for title display
Programme navigation (Still picture)	M	Still picture can be used for title display

Table 10-2 – Recording related functions

Function	Required level	Explanation
Automatic title insertion	M	Title can be given automatically (programme name) in EPG related reservation
EPG related reservation	M	Programme reservation can be done from EPG display
Prolonged sport programme	M	Automatic recording is required for a prolonged portion of a sports programme when it happens. This can be applied when recording a reservation of another programme in the same channel.
Renewal recording	O	Renewal recording function of periodic scheduled programme
Recording reservation	M	At least one week long recording reservation is required by EPG or manual operation.
Repeat recording reservation	M	Periodic reservation of programme is required.

In addition to the recording function, the following editing functions are shown in Table 10-3.

Table 10-3 – Editing functions

Function	Required level	Explanation
Chapter	R	Setting chapter is recommended in recorded programme. Start-up by chapter is recommended.
Title name	M	Editing of title name or thumbnail picture is required.
Title separation	R	Separation at any point is recommended.
Title display	R	Deletion of title display at any time (start, end or intermittent) is recommended.
Move	M	Move to external device is required in accordance with regional broadcast standard.
HD dubbing and move	O	Dubbing and move of digital content are optionally required to external devices without transforming to analogue signal.

Replaying function related to recording is shown in Table 10-4.

Table 10-4 – Replaying function related to recording

Function	Required level	Explanation
Replay	M	Replay of recorded programme. Point of replay stopping by event (operation of stopping button or usage of other function) shall be recorded. Replay in next time shall be started at this point.
Fast forward	M	Fast forward at least in three stages is required.
Play back	M	Play back at least in three stages is required.
Slow replay	O	Replay below original content speed is recommended.
Scene forward and back	O	Scene forward and back are optionally required by frame basis.

Table 10-4 – Replaying function related to recording

Function	Required level	Explanation
Stop once	M	Stop once of replay is required.
Stop	M	Stop of replay is required.
Fast replay	O	Fast replay with approx. 1.3 times speed is optionally required and content shall be audible.
Slow replay	O	Slow replay with approx. 0.9/0.8 times speed is optionally required and content shall be audible.
Time bar	M	Recorded time and present replay point shall be displayed using bar chart.
Instant replay	M	Replay in approx. 30 seconds, then move to replay mode.
Instant skip	M	Fast forward in approx. 30 seconds, then move to replay mode.
Play list	O	Gathering recorded programmes and continuous replay is optionally required.
Time slip	M	Replay including trick play shall be required in recorded programme.

10.3 Installation assistance

For STB installation, assistance in the following function shall be required:

- STB setup, including favourite channel setting, parental rating control, picture aspect ratio (4:3/16:9), STB password, maintenance information for automatic testing, title on/off setting, regional number, etc..
- Installation aid, including model name, ID and software version of STB, NIT frequency and network ID setting for terrestrial broadcast, satellite broadcast and CATV, cable modem information (status, event log, up and down-stream information), IP network information (MAC address, IP address subnet mask, default gateway, etc.).
- The installation process, including description format, simple installation mode, and manual setting.
- STB control from the head-end, including the channel scan function from head-end, and network ID deletion function.

Annex A

Detail of audience measurement for hybrid cable STB

(This annex forms an integral part of this Recommendation.)

A.1 Delivery sequence

In this Recommendation, two types of files are used for audience measurement: the audience measurement configuration file and the measurement report file. The former file conveys a configuration package (defined in [b-ITU-T H.741.2]) to the STB, and the latter is the file format for each measurement report (defined in [b-ITU-T H.741.2]). In this section, the delivery sequence of these two files is described.

A.1.1 Initial setting

As described in clause 7.2.2.1, the hybrid cable STB uses [BBF TR-069] as the transmission protocol for audience measurement messages. Therefore, the STB shall acquire the audience measurement settings, including the report interval, measurement items from [BBF TR-069] the ACS (auto configuration server) during the initial setup of [BBF TR-069]. Figure A.1 shows the processing sequence of the initial setup.

As shown in the sequence, the FileType "**X OUI AudienceMeasurement_config**" indicates the audience measurement configuration file.

[NOTE] – The name of the operator, manufacturer, or organization should be used as "OUI" of the filetype.

A.1.1.1 Setting of audience measurement report interval

In [BBF TR-069], the periodic access interval from the user terminal to the ACS is defined as the parameter named:

Device.ManagementServer.PeriodicInformInterval. Corresponding to this parameter, the user terminal will access the ACS periodically. When the ACS uses the **Device.ManagementServer.PeriodicInformInterval** as the report interval for audience measurement, the STB shall send the audience measurement report at this interval. However, if the ACS indicates the report interval separately, the STB shall read the interval parameter specified in [BBF TR-069] and shall adopt it.

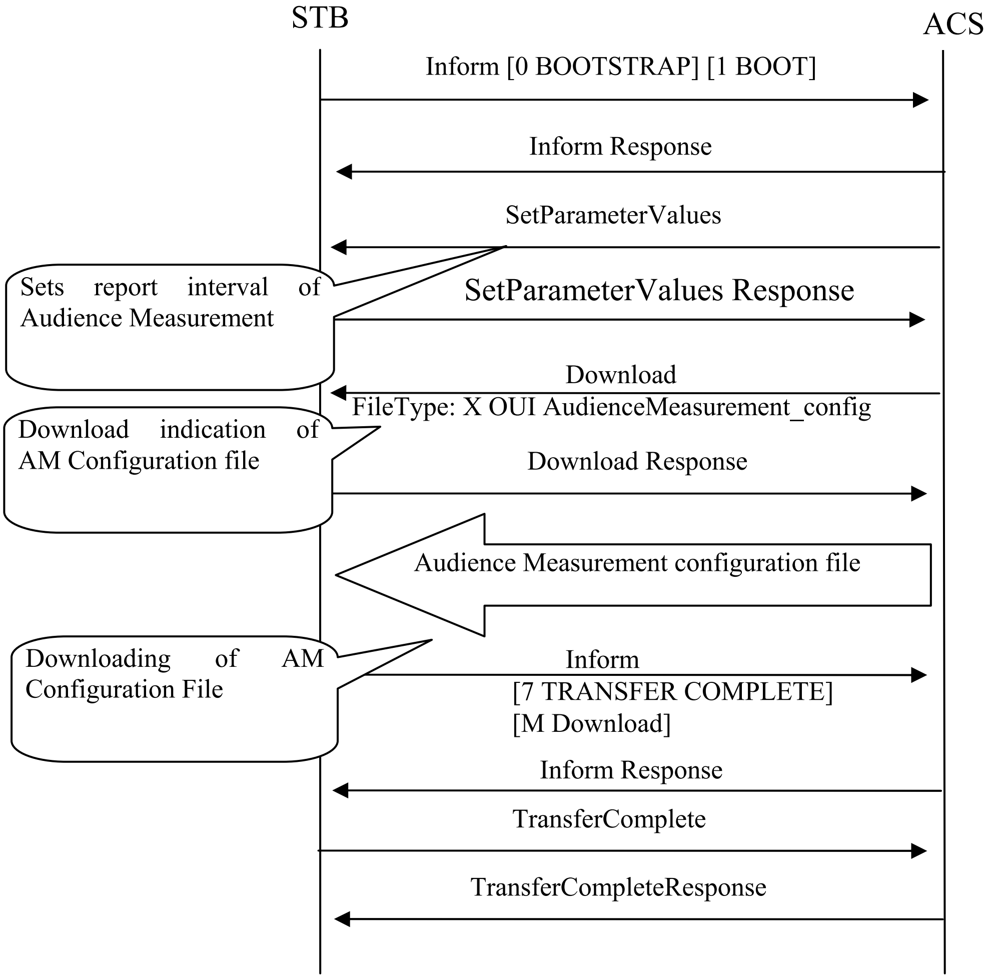

STB
ACS
Inform [0 BOOTSTRAP] [1 BOOT]
Inform Response
SetParameterValues
Sets report interval of Audience Measurement
SetParameterValues Response
Download
FileType: X OUI AudienceMeasurement_config
Download indication of AM Configuration file
Download Response
Audience Measurement configuration file
Downloading of AM Configuration File
Inform
[7 TRANSFER COMPLETE]
[M Download]
Inform Response
TransferComplete
TransferCompleteResponse

Figure A.1 – Initial sequence of audience measurement setting

A.1.1.2 Downloading of audience measurement configuration file

The audience measurement configuration file, defined in [ITU-T H.741.0], is a configuration file that indicates the measurement items and measurement terms. The STB shall download and adopt this file in accordance with the indication of the ACS.

A.1.2 Transmission of measurement report

The audience measurement report file contains all measured information from the STB. In this Recommendation, Delayed Push and Pull mode defined in [ITU-T H.741.0] is applied as the transmission model of the report. Therefore, the STB sends the reports at the following two timings:

– A periodic report corresponding to the scheduled interval (Push)

– A report whenever the STB receives a one- time request from the ACS (Pull).

A.1.2.1 Periodic report corresponding to the scheduled interval

When the report interval described in clause A.1.1.1 is set, the STB shall send the audience measurement report file at the specified interval. Figure A.2 shows the sequence. As shown in the sequence, the Inform message for this periodic report shall use "**X OUI_AMUpload**" event code, and FileType "**X OUI AudienceMeasurement_log**" indicates the audience measurement report file.

[NOTE] – The name of the operator, manufacturer, or organization should be used as "OUI" of the event code and the filetype.

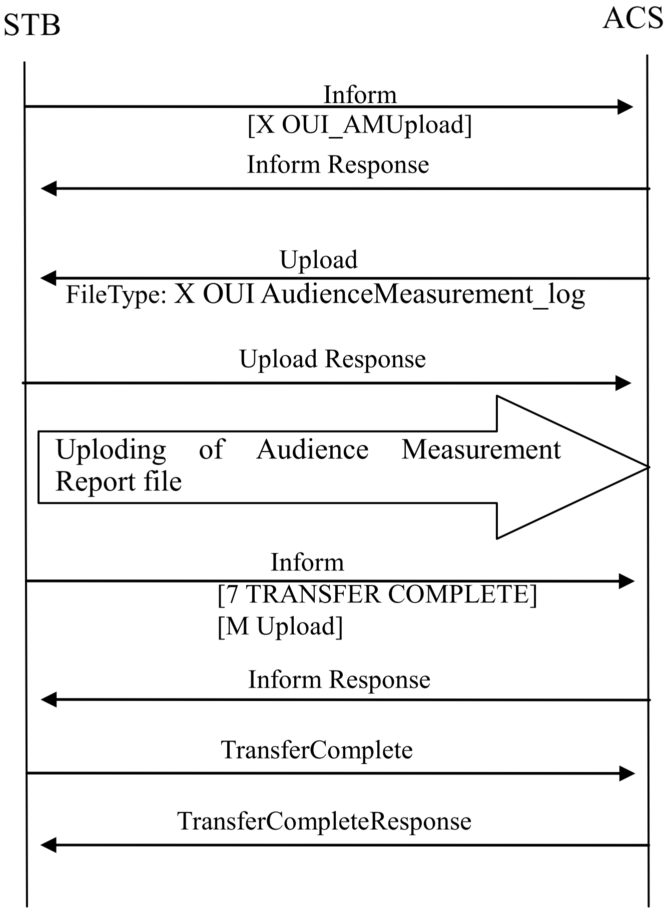


Figure A.2 – Periodic audience measurement report

A.1.2.2 Report corresponding to a request from the ACS

Whenever the STB receives the [BBF TR-069] upload request with the FileType "**X OUI AudienceMeasurement_log**", the STB shall send the audience measurement report file at that time. Figure A.3 shows the sequence.

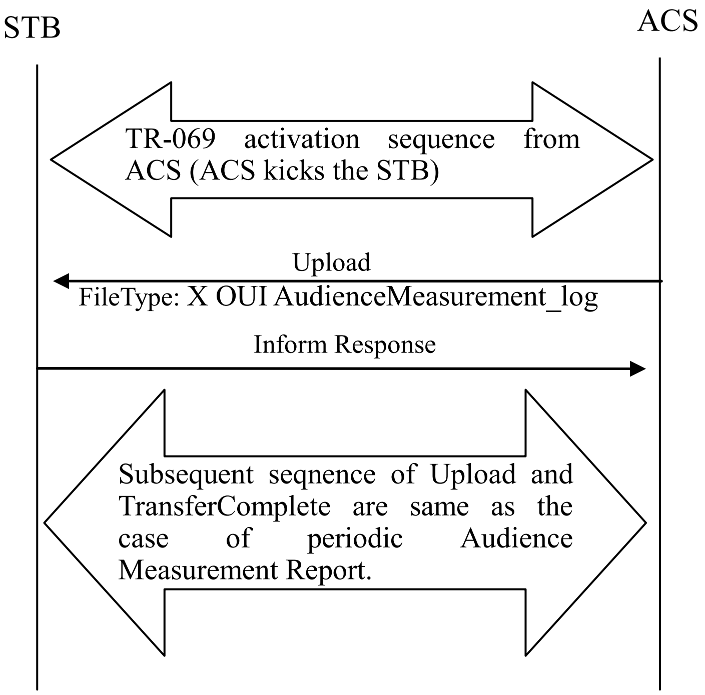

STB
ACS
TR-069 activation sequence from ACS (ACS kicks the STB)
Upload
FileType: X OUI AudienceMeasurement_log
Inform Response
Subsequent seqnence of Upload and TransferComplete are same as the case of periodic Audience Measurement Report.

Figure A.3 – Measurement report sequence corresponding to the specific request

A.1.3 Update of audience measurement configuration

When the operator changes the report interval and report items, the operator will update them through the ACS. The STB shall receive and adopt the specified changes as follows, (see Figure A.4):

A.1.3.1 Measurement interval change

As described in clause A.1.1.1, the interval is defined as the [BBF TR-069] parameter. Therefore, the change is indicated by the SetParameterValue method of [BBF TR-069]. The STB shall receive it and adopt the specified interval as the periodic report interval.

A.1.3.2 Measurement item change

As described in clause A.1.1.2, the audience measurement items are indicated by the audience measurement configuration file. Therefore, when the ACS indicates the download of a new configuration file, the STB shall download and adopt it.

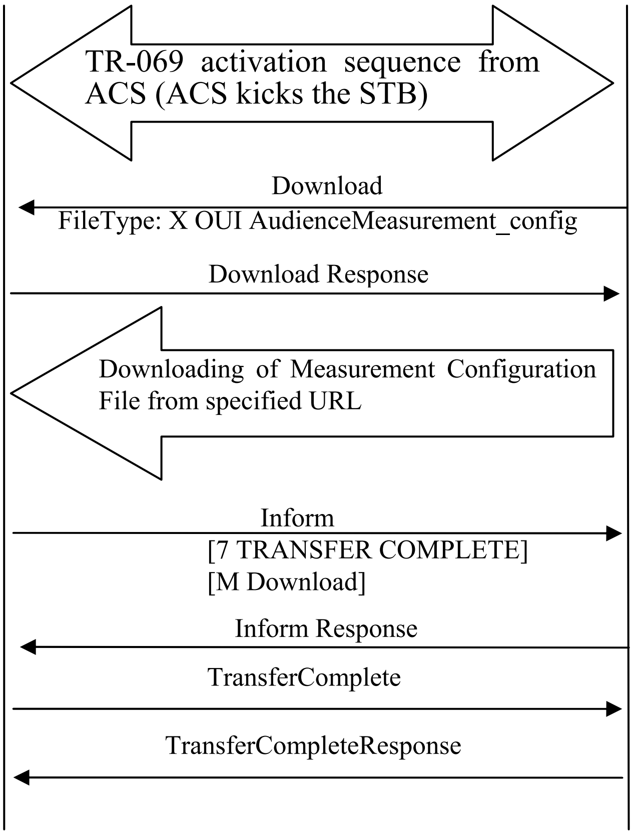

TR-069 activation sequence from ACS (ACS kicks the STB)
Download
FileType: X OUI AudienceMeasurement_config
Download Response
Downloading of Measurement Configuration File from specified URL
Inform
[7 TRANSFER COMPLETE]
[M Download]
Inform Response
TransferComplete
TransferCompleteResponse

Figure A.4 – Update of audience measurement configuration file

Appendix I

Supplemental information on audience measurement messages

(This appendix does not form an integral part of this Recommendation.)

I.1 Additional audience measurement items

In addition to Table 7-7, the audience measurement items in Table I.1 are defined for the hybrid cable STB.

Table I.1 – Audience measurement items

Items	Description	Assumed implementation
UserSettingInfo	This element is a container for the various types of setting information of STB.	Child element of MeasurementReport element.
PermissionLevel	This element indicates current user's permission level (see clause 7.6.5).	Child element of UserSettingInfo element.
OperatorID	This element indicates the cable TV operator which provides the service.	Child element of UserSettingInfo element.
ValidChannel	This element indicates the list of channels which STB can receive.	Child element of UserSettingInfo element
UserPreferences	This element is a container for the user's preference information. It includes the user's rating information (5 to 1) for each genre of the content.	Child element of UserSettingInfo element
UISetting	This element is a container for UI setting information of the STB.	Child element of UserSettingInfo element
Recording event (RecStartEvent, RecStopEvent, RecFailureEvent)	This element informs the start, stop and failure of the recording.	Child element of MeasurementReport element.
StoredContentEvent	This element informs events related to the stored content, such as playback, pause, stop, fast-forward, rewind, deletion of the content, copy of the content, and move of the content.	Child element of MeasurementReport element.
PowerOnEvent /PowerOff Event	This element informs power on and off of the STB.	Child element of Measurement Report element.
Recording Schedule List (RecScheduleList)	This is the list of programmed recording which is valid at the report timing. Content or channel information, recording schedule information, and periodic recording information will be included.	Child element of AMReportPackage element.
Referrer	When any application causes the channel watching, recording, and/or registration of the programmed recording, this attribute provides information of it because it is the contributing factor of the event.	This is the additional attribute of StartNavMethod^{*1}. This is also used to the Navi Method of Recording event and Recording Schedule List. ^{*1}: Defined in [b-ITU-T H.741.3].

I.2 The relationship between the audience measurement capability defined in [b-ITU-T H.741.1] and the capability of the hybrid cable STB

Table I.2 shows the relationship between the audience measurement capability defined in [b-ITU-T H.741.1] and the capability of the advanced cable STB, defined in this Recommendation.

Table I.2 – Audience measurement capability

Capability area	Options defined in [b-ITU-T H.741.1]	Defined in this Recommendation
Transport protocols	Specific protocols are out of scope of AM	TR-069
Transport delivery mode	Unicast	Yes
	Multicast	N.A
Permission mode	External	No
	Internal	Yes[a)]
	Hybrid	Yes[a)]
Configuration mode	Push	Yes
	Pull	N.A
	Hybrid	N.A
Measurement triggers	Event	Yes
	Time sampling	N.A
	Service start sampling	N.A
Report delivery mode	Immediate Push	N.A
	Delayed Push	N.A
	Pull	N.A
	Delayed Push and Pull	Yes
Operational management	Acknowledgements	N.A
	Multicast threshold ranges	N.A
	Multicast error reporting	N.A
	Content Filtering	N.A

[a)] When the user permission information is only directly transferred from the STB to the aggregation function (TR-069 ACS), it is an "internal" permission mode. On the other hand, if the user's permission information can be controlled by other servers (such as the subscriber information server), it is "hybrid" permission mode.

I.3 Schema example of measurement report

Figure I.1 shows the schema example of measurement report. This schema contains all the items described in clause I.1.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
targetNamespace="http://www.jlabs.or.jp/2012/XMLSchema" xmlns:jam="http://www.jlabs.or.jp/2012/XMLSchema"
   xmlns:ns1="urn:itut:iptv:am1:2010" xmlns:ns2="urn:itut:iptv:am:mpeg7:2010">
   <xs:element name="AMReportPackage">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="jam:SubscriberID" maxOccurs="1" minOccurs="0">
               <xs:annotation>
                  <xs:documentation>String[255]</xs:documentation>
               </xs:annotation>
            </xs:element>
            <xs:element ref="jam:TerminalDeviceID"/>
            <xs:element name="MeasurementReport" maxOccurs="unbounded" minOccurs="0">
               <xs:annotation>
                  <xs:documentation>One MeasurmentReport contains one event.</xs:documentation>
               </xs:annotation>
               <xs:complexType>
                  <xs:choice>
                     <xs:element ref="jam:DeviceInformation">
                        <xs:annotation>
                           <xs:documentation>Except the case of PermissionLevel 0, one Audience Measurement File must
contain one DeviceInformation.</xs:documentation>
                        </xs:annotation>
                     </xs:element>
                     <xs:element ref="jam:UserSettingInfo">
                        <xs:annotation>
                           <xs:documentation>Except the case of PermissionLevel 0, one Audience Measurement File must
contain one UserSettingInfo.</xs:documentation>
                        </xs:annotation>
                     </xs:element>
                     <xs:element name="ChannelStartEvent" minOccurs="0">
                        <xs:annotation>
                           <xs:documentation>Indicates that a channel start event has taken place.</xs:documentation>
                        </xs:annotation>
                        <xs:complexType>
                           <xs:complexContent>
                              <xs:extension base="jam:LinearTVEventType">
                                 <xs:sequence>
                                    <xs:element ref="jam:StartNavMethod"/>
                                    <xs:element ref="jam:ChannelInfo"/>
                                    <xs:element minOccurs="0" ref="jam:ViewMode"/>
                                 </xs:sequence>
                              </xs:extension>
                           </xs:complexContent>
                        </xs:complexType>
                     </xs:element>
                     <xs:element name="ChannelStopEvent" minOccurs="0">
                        <xs:complexType>
                           <xs:complexContent>
                              <xs:extension base="jam:LinearTVEventType">
                                 <xs:sequence>
                                    <xs:element ref="jam:StopNavMethod"/>
                                 </xs:sequence>
                              </xs:extension>
                           </xs:complexContent>
                        </xs:complexType>
                     </xs:element>
                     <xs:element name="RecStartEvent" minOccurs="0">
                        <xs:annotation>
                           <xs:documentation>Indicates that a content recoding event has taken place.</xs:documentation>
                        </xs:annotation>
                        <xs:complexType>
                           <xs:complexContent>
                              <xs:extension base="jam:RecEventType">
                                 <xs:sequence>
                                    <xs:element ref="jam:StartRecMethod"/>
                                    <xs:element ref="jam:ChannelInfo"/>
                                    <xs:element name="TargetDeviceLocation" type="jam:RecLocation">
                                       <xs:annotation>
                                          <xs:documentation>int 0 - 63 0 = built-in HDD 1 = external HDD connected by
USB 2 = Other in-home device (DLNA uploading recording) 63 =
                                             unknown</xs:documentation>
                                       </xs:annotation>
                                    </xs:element>
                                 </xs:sequence>
                                 <xs:attribute ref="jam:ScheduleID"/>
                              </xs:extension>
                           </xs:complexContent>
                        </xs:complexType>
                     </xs:element>
                     <xs:element name="RecStopEvent" minOccurs="0">
                        <xs:complexType>
                           <xs:complexContent>
                              <xs:extension base="jam:RecEventType">
                                 <xs:sequence>
                                    <xs:element ref="jam:StopRecMethod"/>
                                    <xs:element maxOccurs="1" minOccurs="0" ref="jam:ErrorCode"/>
                                 </xs:sequence>
                              </xs:extension>
                  </xs:complexContent>
                </xs:complexType>
              </xs:element>
              <xs:element minOccurs="0" name="RecFailureEvent">
                <xs:complexType>
                  <xs:complexContent>
                    <xs:extension base="jam:RecEventType">
                      <xs:sequence>
                        <xs:element ref="jam:ErrorCode"/>
                      </xs:sequence>
                    </xs:extension>
                  </xs:complexContent>
                </xs:complexType>
              </xs:element>
              <xs:element name="StoredContentEvent" type="jam:StoredContentEventType" minOccurs="0">
                <xs:annotation>
                  <xs:documentation>This event indicates any operation such as Playback, playback, pause, stop,
fast-forward, rewind, deletion of the content, copy of the content, and
                    move of the content.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element ref="jam:DisplayStatus" minOccurs="0"/>
              <xs:element minOccurs="0" name="PowerOnEvent" type="jam:PowerOnOffEventType"
maxOccurs="1"/>
              <xs:element minOccurs="0" name="PowerOffEvent" type="jam:PowerOnOffEventType"
maxOccurs="1"/>
            </xs:choice>
            <xs:attribute ref="jam:MeasurementRequestID"/>
            <xs:attribute name="MeasurementReportTriggerTime" type="xs:dateTime" use="required">
              <xs:annotation>
                <xs:documentation>The time of each event.</xs:documentation>
              </xs:annotation>
            </xs:attribute>
          </xs:complexType>
        </xs:element>
        <xs:element ref="jam:RecScheduleList" minOccurs="0">
          <xs:annotation>
            <xs:documentation>List of programmed recording which is valid at the report timing. Content or
channel information, recording schedule information, and periodic recording
              information will be included.</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="MeasurementReportCreationSenderTime" use="required" type="xs:dateTime">
        <xs:annotation>
          <xs:documentation>Sender time of the report.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:element name="TVInformation">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" name="TVManuf" type="xs:hexBinary">
          <xs:annotation>
            <xs:documentation>HexBinaly 2 byte, Vender ID of TV is acquired through the Plag and Play
information.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element minOccurs="0" name="TVModel" type="xs:hexBinary">
          <xs:annotation>
            <xs:documentation>HexBinaly 2 byte</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element maxOccurs="1" minOccurs="0" name="TVSerialNum" type="xs:hexBinary">
          <xs:annotation>
            <xs:documentation>HexBinaly 4 byte</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="STBInformation">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="1" name="STBManuf" type="xs:hexBinary">
          <xs:annotation>
            <xs:documentation>HexBinaly 3 Byte, OUI of the STB</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element minOccurs="1" name="STBModel" type="xs:string">
          <xs:annotation>
            <xs:documentation/>
          </xs:annotation>
        </xs:element>
        <xs:element maxOccurs="1" minOccurs="1" name="STBSerialNum" type="xs:hexBinary">
          <xs:annotation>
            <xs:documentation>HexBinaly 6 byte</xs:documentation>
          </xs:annotation>
        </xs:element>
```

```
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="UserSettingInfo">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="PermissionLevel" type="xs:int">
               <xs:annotation>
                  <xs:documentation>Indicates user's PermissionLevel of audience measurement.</xs:documentation>
               </xs:annotation>
            </xs:element>
            <xs:element name="OperatorID" minOccurs="1" type="xs:string">
               <xs:annotation>
                  <xs:documentation>String[3]. Identifier of the service operator.</xs:documentation>
               </xs:annotation>
            </xs:element>
            <xs:element minOccurs="0" name="ValidChannel">
               <xs:complexType>
                  <xs:sequence>
                     <xs:element name="numOfChannels" type="xs:int">
                        <xs:annotation>
                           <xs:documentation>int 0-255 Num of channels.</xs:documentation>
                        </xs:annotation>
                     </xs:element>
                     <xs:element name="ChannelList" type="xs:hexBinary" maxOccurs="1" minOccurs="0">
                        <xs:annotation>
                           <xs:documentation>String[2040] indicates the list of valid channels as the consecutive strings of
NetworkID (2byte) and serviceID (2byte).</xs:documentation>
                        </xs:annotation>
                     </xs:element>
                  </xs:sequence>
               </xs:complexType>
            </xs:element>
            <xs:element name="UserPreferences" minOccurs="0" maxOccurs="1">
               <xs:complexType>
                  <xs:sequence maxOccurs="unbounded">
                     <xs:element ref="jam:UserPreference"/>
                  </xs:sequence>
               </xs:complexType>
            </xs:element>
            <xs:element minOccurs="0" ref="jam:UISettings"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="UISettings">
      <xs:annotation>
         <xs:documentation/>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence>
            <xs:element name="UIName">
               <xs:annotation>
                  <xs:documentation>ASCII String[64] e.g.,: com.appdeveloper.HomeUI</xs:documentation>
               </xs:annotation>
               <xs:complexType>
                  <xs:simpleContent>
                     <xs:extension base="xs:string">
                        <xs:attribute name="Version" type="xs:string"/>
                     </xs:extension>
                  </xs:simpleContent>
               </xs:complexType>
            </xs:element>
            <xs:element name="UIType">
               <xs:annotation>
                  <xs:documentation>ASCII String[16]: The identifier of specific UI setting such as "ActiveUI",
"SimpleUI", "2012modelUI", etc.</xs:documentation>
               </xs:annotation>
               <xs:complexType>
                  <xs:simpleContent>
                     <xs:extension base="xs:string"> </xs:extension>
                  </xs:simpleContent>
               </xs:complexType>
            </xs:element>
            <xs:element name="MyChannelSetting">
               <xs:annotation>
                  <xs:documentation>ASCII String[16]: Whether the STB is "Favorite Channel filter mode" or not. The
returning value must "Enable", "Disable", or "Unknown".</xs:documentation>
               </xs:annotation>
               <xs:complexType>
                  <xs:simpleContent>
                     <xs:extension base="xs:string">
                        <xs:attribute name="MyChannelsInfo" type="xs:hexBinary" use="optional">
                           <xs:annotation>
                              <xs:documentation>String[400]: Channel list of user's favorite channel filter. (Up to 50
channels.)</xs:documentation>
                           </xs:annotation>
                        </xs:attribute>
                     </xs:extension>
                  </xs:simpleContent>
               </xs:complexType>
            </xs:element>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="DeviceInformation">
      <xs:complexType>
         <xs:sequence>
            <xs:annotation>
               <xs:documentation/>
            </xs:annotation>
            <xs:element ref="jam:TVInformation" minOccurs="0"/>
            <xs:element ref="jam:STBInformation"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="SubscriberID" type="jam:UserID">
      <xs:annotation>
         <xs:documentation/>
      </xs:annotation>
   </xs:element>
   <xs:simpleType name="UserID">
      <xs:annotation>
         <xs:documentation>String[255]</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string"/>
   </xs:simpleType>
   <xs:element name="TerminalDeviceID" type="xs:string">
      <xs:annotation>
         <xs:documentation>MAC Address of the STB</xs:documentation>
      </xs:annotation>
   </xs:element>
   <xs:element name="ControlDevice">
      <xs:annotation>
         <xs:documentation>int 0 - 63 0 = Remote 1 = STB Front Panel 33 = System event (Non user input) 34 = Any
network devices (PC, Tablet, Mobile, etc.) 63 = Unknown</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:simpleContent>
            <xs:extension base="xs:int"> </xs:extension>
         </xs:simpleContent>
      </xs:complexType>
   </xs:element>
   <xs:element name="DisplayStatus">
      <xs:annotation>
         <xs:documentation>int 0-63 0 = STB's output is displayed to any display device (mainly TV). 1 = STB's output
is not displayed to any display device. 63 = Display status is
            unknown.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:simpleContent>
            <xs:extension base="xs:int"> </xs:extension>
         </xs:simpleContent>
      </xs:complexType>
   </xs:element>
   <xs:element name="ChannelInfo">
      <xs:annotation>
         <xs:documentation/>
      </xs:annotation>
      <xs:complexType>
         <xs:attribute name="NetworkID" type="xs:hexBinary" use="required">
            <xs:annotation>
               <xs:documentation>2Byte HexBinary 0x0000 - 0xffff</xs:documentation>
            </xs:annotation>
         </xs:attribute>
         <xs:attribute name="ServiceID" type="xs:hexBinary" use="required">
            <xs:annotation>
               <xs:documentation>2Byte HexBinary 0x0000 - 0xffff</xs:documentation>
            </xs:annotation>
         </xs:attribute>
         <xs:attribute name="ShortName" type="xs:string">
            <xs:annotation>
               <xs:documentation>String[20]</xs:documentation>
            </xs:annotation>
         </xs:attribute>
      </xs:complexType>
   </xs:element>
   <xs:element name="RecSchedule">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="ScheduledTime">
               <xs:annotation>
                  <xs:documentation>Target date and time of the programmed recording.</xs:documentation>
               </xs:annotation>
               <xs:complexType>
                  <xs:sequence>
                     <xs:element minOccurs="0" ref="jam:DayOfTheWeek">
                        <xs:annotation>
                           <xs:documentation/>
                        </xs:annotation>
                     </xs:element>
                     <xs:element name="From" type="xs:date" maxOccurs="0" minOccurs="0"/>
                     <xs:element name="Until" type="xs:date" maxOccurs="0" minOccurs="0"/>
                  </xs:sequence>
                  <xs:attributeGroup ref="jam:ScheduledTime"/>
               </xs:complexType>
            </xs:element>
            <xs:element ref="jam:ChannelInfo">
               <xs:annotation>
                  <xs:documentation>Channel information of the content</xs:documentation>
               </xs:annotation>
            </xs:element>
            <xs:element maxOccurs="99" minOccurs="0" ref="jam:ContentInfo">
               <xs:annotation>
                  <xs:documentation>Content information of programmed recording. If the recording is set by the pair of
starting time and ending time, this information will not be exist.
                     If the recording is set by Content ID, this information should be exist.</xs:documentation>
               </xs:annotation>
            </xs:element>
            <xs:element ref="jam:RecScheduleMethod"/>
         </xs:sequence>
         <xs:attribute ref="jam:ScheduleID"/>
         <xs:attribute name="Time" type="xs:dateTime" use="required">
            <xs:annotation>
               <xs:documentation>Date and time information at which user adds the scheduled
recording.</xs:documentation>
            </xs:annotation>
         </xs:attribute>
      </xs:complexType>
   </xs:element>
   <xs:complexType name="LinearTVEventType">
      <xs:attribute ref="jam:ServiceInstanceID"/>
   </xs:complexType>
   <xs:attribute name="ReferrerAppID" type="xs:string">
      <xs:annotation>
         <xs:documentation>ASCII String[64] When LocalApp or RemoteApp is set as the NavMethod, the
application's ID should be set to this attribute. e.g.,:
            com.hogehoge.foo.epgapp.</xs:documentation>
      </xs:annotation>
   </xs:attribute>
   <xs:attribute name="ReferrerScreenID" type="xs:string">
      <xs:annotation>
         <xs:documentation>ASCII String[64] The identifier of the screen which causes the event. e.g: TopPage,
SearchResult, EPGGrid </xs:documentation>
      </xs:annotation>
   </xs:attribute>
   <xs:attribute name="ReferrerAppDomain" type="xs:string">
      <xs:annotation>
         <xs:documentation>ASCII String[64]</xs:documentation>
      </xs:annotation>
   </xs:attribute>
   <xs:attribute name="ReferrerAppVer" type="xs:string">
      <xs:annotation>
         <xs:documentation>ACSII String[64] e.g: 1.1, 1.1.23RC </xs:documentation>
      </xs:annotation>
   </xs:attribute>
   <xs:attributeGroup name="Referrer">
      <xs:attribute ref="jam:ReferrerAppID"/>
      <xs:attribute ref="jam:ReferrerScreenID"/>
      <xs:attribute ref="jam:ReferrerAppDomain"/>
      <xs:attribute ref="jam:ReferrerAppVer"/>
   </xs:attributeGroup>
   <xs:element name="ViewMode" type="xs:int">
      <xs:annotation>
         <xs:documentation>int 0 - 63 0 = not displayed 1 = full-screen 2 = Picture-In-Picture 63 =
unknown</xs:documentation>
      </xs:annotation>
   </xs:element>
   <xs:complexType name="RecEventType">
      <xs:attribute ref="jam:ServiceInstanceID"/>
   </xs:complexType>
   <xs:element name="StartNavMethod">
      <xs:annotation>
         <xs:documentation>int 0 - 63 0 = Up/Down arrow, etc. 1 = Channel surfing 2 = Scheduled channel change
(Programmed channel change, etc.) 3 = EPG 4 = Last channel 6 = Specified channel input
            method (e.g. 3 digit input, preset channel input) 32 = Channel change from local (running on the
STB)application 33 = Channel change from remote (running on the smart phone, tablet,
            etc.) application 63 = unknown</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:simpleContent>
            <xs:extension base="xs:int">
               <xs:attributeGroup ref="jam:Referrer"/>
            </xs:extension>
         </xs:simpleContent>
      </xs:complexType>
   </xs:element>
   <xs:element name="StopNavMethod">
      <xs:annotation>
         <xs:documentation>int 0 - 63 0 = Up/Down arrow, etc. 1 = Channel surfing 2 = Scheduled channel change
(Programmed channel change, etc.) 3 = EPG 6 = Specified channel input method (e.g., 3
        digit input, preset channel input) 32 = Channel change from local (running on the STB) application 33 =
Channel change from remote (running on the smart phone, tablet, etc.) application
        61 = Other reason 62 = Channel stop caused by power off 63 = unknown</xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:element name="StartRecMethod">
    <xs:annotation>
      <xs:documentation>int 0 - 63 0 = Instant recording start caused by the user's direct input 1 = Scheduled
recording start 4 = Recording start based on the content recommendation 32 =
        Channel recording caused by local (running on the STB) application 33 = Channel recording caused by
remote (running on the smart phone, tablet, etc.) application 63= unknown
      </xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:int">
          <xs:attributeGroup ref="jam:Referrer"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>
  <xs:element name="StopRecMethod" type="xs:int">
    <xs:annotation>
      <xs:documentation>int 0 - 63 0 = Error 1 = Recording is completed 2 = Recording is terminated by user/system
operation 63= unknown</xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:simpleType name="RecLocation">
    <xs:annotation>
      <xs:documentation/>
    </xs:annotation>
    <xs:restriction base="xs:int"/>
  </xs:simpleType>
  <xs:attribute name="ServiceInstanceID" type="xs:int">
    <xs:annotation>
      <xs:documentation>int 0-63 0: No tuner used 1-50: Identifier of the tunner 51-62: Reserved 63: unknown
</xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:complexType name="ScheduleChangeEventType">
    <xs:sequence>
      <xs:element ref="jam:ControlDevice"/>
    </xs:sequence>
    <xs:attribute name="Time" type="xs:dateTime" use="required"/>
    <xs:attribute name="Priority" type="xs:int">
      <xs:annotation>
        <xs:documentation>int 0-100 Priority of recording set by user or STB. </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:attributeGroup name="ScheduledTimeOfDay">
    <xs:attribute name="StartTime" type="xs:time"/>
    <xs:attribute name="EndTime" type="xs:time"/>
  </xs:attributeGroup>
  <xs:element name="ContentInfo">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="jam:ContentTitle"/>
        <xs:element maxOccurs="3" minOccurs="0" ref="jam:Genre"/>
      </xs:sequence>
      <xs:attribute name="ContentID" type="xs:normalizedString" use="optional">
        <xs:annotation>
          <xs:documentation>Program ID specified by the recording/scheduled recording.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:element name="StoredContentInfo">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="RecordedDate" minOccurs="1">
          <xs:annotation>
            <xs:documentation>Date and time information at which the content is recorded.</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:attributeGroup ref="jam:StoredContentTime"/>
          </xs:complexType>
        </xs:element>
        <xs:element minOccurs="0" ref="jam:ContentTitle">
          <xs:annotation>
            <xs:documentation/>
          </xs:annotation>
        </xs:element>
        <xs:element maxOccurs="3" minOccurs="0" ref="jam:Genre"/>
      </xs:sequence>
      <xs:attribute name="ContentID" type="xs:hexBinary" use="optional">
        <xs:annotation>
          <xs:documentation>Program ID specified by the recording/scheduled recording.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="StoredContentEventType">
    <xs:sequence>
      <xs:element ref="jam:ChannelInfo" minOccurs="1"/>
      <xs:element ref="jam:StoredContentInfo">
        <xs:annotation>
          <xs:documentation/>
        </xs:annotation>
      </xs:element>
      <xs:sequence maxOccurs="unbounded">
        <xs:element name="StoredContentOperation" type="jam:Operation">
          <xs:annotation>
            <xs:documentation>Identifies to indicates the user operation. PLAY PAUSE FF RW STOP DEL COPY
MOVE </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:sequence>
  </xs:complexType>
  <xs:attributeGroup name="ScheduledTime">
    <xs:attribute name="Start" type="xs:dateTime" use="required">
      <xs:annotation>
        <xs:documentation/>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="Duration" type="xs:duration" use="required">
      <xs:annotation>
        <xs:documentation>e.g., PT1H59M59S, PT2H</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:attributeGroup>
  <xs:attributeGroup name="ContentAbsoluteTime">
    <xs:annotation>
      <xs:documentation>Elapsed time from the beginning of the content. (The begening of the content is
00:00:00.)</xs:documentation>
    </xs:annotation>
    <xs:attribute name="time" type="xs:time"/>
  </xs:attributeGroup>
  <xs:complexType name="Operation">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attributeGroup ref="jam:ContentAbsoluteTime"/>
        <xs:attribute name="Scale" type="xs:float" use="optional">
          <xs:annotation>
            <xs:documentation/>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
  <xs:element name="UserPreference">
    <xs:annotation>
      <xs:documentation>Indicates the user's preference to specified genre. (5: Very favorable, 4: Favorable, 3:
Neutral, 2: Unfavorable, 1: Very unfavorable)</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:int">
          <xs:attribute ref="jam:GenreLevel1"/>
          <xs:attribute ref="jam:GenreLevel2"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>
  <xs:attributeGroup name="DeliveryWindowTime">
    <xs:attribute name="StartDeliveryWindowTime" type="xs:time"/>
    <xs:attribute name="EndDeliveryWindowTime" type="xs:time"/>
  </xs:attributeGroup>
  <xs:element name="DayOfTheWeek">
    <xs:complexType>
      <xs:attribute name="All" type="xs:boolean">
        <xs:annotation>
          <xs:documentation/>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="Sun" type="xs:boolean"/>
      <xs:attribute name="Mon" type="xs:boolean"/>
      <xs:attribute name="Tue" type="xs:boolean"/>
      <xs:attribute name="Wed" type="xs:boolean"/>
      <xs:attribute name="Thu" type="xs:boolean"/>
      <xs:attribute name="Fri" type="xs:boolean"/>
      <xs:attribute name="Sat" type="xs:boolean"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="RecScheduleList">
    <xs:complexType>
      <xs:sequence maxOccurs="unbounded">
        <xs:element ref="jam:RecSchedule"/>
      </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="RecScheduleMethod">
      <xs:annotation>
         <xs:documentation>Indicates the method used to schedule addition. int 0 - 63 1 = Recording menu 3 =
Automatic scheduling based on the user's keyword 4 = Automatic scheduling without user's
            input. (Automatic recommendation, etc.) 32 = LocalApp: Scheduling from local (running on the
STB)application 33 = RemoteApp: Scheduling from remote (running on the smart phone, tablet,
            etc.) application 63= unknown</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:simpleContent>
            <xs:extension base="xs:int">
               <xs:attributeGroup ref="jam:Referrer"/>
            </xs:extension>
         </xs:simpleContent>
      </xs:complexType>
   </xs:element>
   <xs:attribute name="ScheduleID" type="xs:hexBinary">
      <xs:annotation>
         <xs:documentation>HexBinary 2 byte Identifier of each recording shedule.</xs:documentation>
      </xs:annotation>
   </xs:attribute>
   <xs:complexType name="PowerOnOffEventType">
      <xs:sequence>
         <xs:element ref="jam:ControlDevice"/>
      </xs:sequence>
   </xs:complexType>
   <xs:attribute name="Start" type="xs:dateTime"/>
   <xs:attributeGroup name="StoredContentTime">
      <xs:attribute ref="jam:Start" use="required"/>
      <xs:attribute name="duration" type="xs:duration" use="required">
         <xs:annotation>
            <xs:documentation>e.g.,: PT1H59M59S, PT2H</xs:documentation>
         </xs:annotation>
      </xs:attribute>
   </xs:attributeGroup>
   <xs:element name="ErrorCode" type="xs:int">
      <xs:annotation>
         <xs:documentation>int 0-65535</xs:documentation>
      </xs:annotation>
   </xs:element>
   <xs:element name="ContentTitle" type="xs:string">
      <xs:annotation>
         <xs:documentation>String[80]</xs:documentation>
      </xs:annotation>
   </xs:element>
   <xs:element name="Genre">
      <xs:annotation>
         <xs:documentation/>
      </xs:annotation>
      <xs:complexType>
         <xs:attribute ref="jam:GenreLevel1"/>
         <xs:attribute ref="jam:GenreLevel2"/>
      </xs:complexType>
   </xs:element>
   <xs:attribute name="GenreLevel1" type="xs:string">
      <xs:annotation>
         <xs:documentation>String[1]</xs:documentation>
      </xs:annotation>
   </xs:attribute>
   <xs:attribute name="GenreLevel2" type="xs:string">
      <xs:annotation>
         <xs:documentation>String[1]</xs:documentation>
      </xs:annotation>
   </xs:attribute>
   <xs:attribute name="MeasurementRequestID" type="xs:int">
      <xs:annotation>
         <xs:documentation>int 0-255</xs:documentation>
      </xs:annotation>
   </xs:attribute>
</xs:schema>
```

Figure I.1 – Example of schema of the audience measurement report

Appendix II

Required BBF TR-069 profiles

(This appendix does not form an integral part of this Recommendation.)

II.1 Data model profile and version number

The following profiles are required for the implementation of [BBF TR-069] in the hybrid cable set-top box. The number in brackets indicates the version number of the data model which describes the profile.

TR-157 Amendment 3

- Baseline:1 (1.0)
- GatewayInfo:1 (1.0, WAN-LAN)
- Time:1 (1.0)
- LAN:1 (1.0)
- IPPing:1 (1.0)
- TraceRoute:1 (1.0)
- Download:1 (1.2)
- Upload:1 (1.2)
- MemoryStatus:1 (1.3)
- ProcessStatus:1 (1.3)
- TempStatus:1 (1.3)
- AutonXferComplPolicy:1 (1.3)
- UPnPDev:1 (1.3)
- UPnPDiscBasic:1 (1.3)
- UPnPDiscAdv:1 (1.3)
- SelfTestDiag:1 (1.3)
- NSLookupDiag:1 (1.3)
- SimpleFirewall:1 (1.3)
- USBHostsBasic:1 (1.3)
- USBHostsAdv:1 (1.3)
- PeriodicStatsBase:1 (1.3)
- PeriodicStatsAdv:1 (1.3)
- DownloadAnnounce:1 (1.3)
- DownloadQuery:1 (1.3)
- Processors:1 (1.7)
- VendorLogFiles:1 (1.7)
- DUStageChngComplPolicy:1 (1.7)
- SM_ExecEnvs:1 (1.7)

In addition, it is required that the following objects and parameters be supported:

- Device.DLNA.
- Device.SmartCardReader.{i}.

TR-135 Amendment 1 (Informative)

- Baseline:1 (1.0)
- PVR:1 (1.0)
- DTT:1 (1.0)
- IPTVBaseline:1 (1.0)
- RTCP:1 (1.0)
- RTPAVPF:1 (1.0)
- RTPAVConfig (1.1)
- FECConfig:1 (1.1)
- ForceMonitoring:1 (1.1)
- IPTVHomeNetwork:1 (1.0)
- IGMP:1 (1.0)
- BasicPerfMon:1 (1.0)
- BasicPerfMon:2 (1.1)
- ECPerfMon:1 (1.0)
- VideoPerfMon:1 (1.0)
- AudioPerfMon:1 (1.0)
- DiagPerfMon:1 (1.1)
- AudienceStats:1 (1.0)
- AnalogOutput:1 (1.0)
- DigitalOutput:1 (1.0)
- DigitalOutput:2 (1.1)
- CA:1 (1.0)
- DRM:1 (1.0)
- CDS:1 (1.1)

TR-140 Amendment 1

- Baseline:1 (1.0)
- Baseline:2 (1.1)
- VolumeConfig:1 (1.0)
- UserAccess:1 (1.0)
- UserAccess:2 (1.1)
- GroupAccess:1 (1.0)
- GroupAccess:2 (1.1)

Appendix III

Extended Java APIs

(This appendix does not form an integral part of this Recommendation.)

This appendix defines extended Java APIs to provide STB specific functionalities to applications running on the procedural application environment of the hybrid cable STB. This annex only focuses on the extended Java APIs, and does NOT include any standard APIs of Java, or those defined by the operating system.

In this Recommendation, the extended Java APIs are required to be accessible only to the authorized applications deployed by authorities such as cable television operators, regional broadcasting authorities, etc. The "Java-cable" notation in Appendix III is an example of name space given in this document. Other names can be applicable for actual implementation. It is very useful to provide information on change of the name space by implementer, to maintain the interoperability of the application as much as possible.

III.1 Overview

Architecture of the procedural application environment

The position of the procedural application environment APIs among the overall STB architecture is shown in Figure III.1. These APIs enable applications running on the virtual machine, hereinafter referred to as "procedural applications", to call various functions implemented in the STB, such as hardware, library software, etc. The API definitions focus only on the interface specifications, such as class methods, events, inter-process communications, etc., of the extended Java APIs, which can be seen by the procedural application side. This means that the API definitions will never specify the way of the implementations. For example, a class can be implemented as either a programme running on the virtual machine or a programme utilizing the lower layer libraries through the Java Native Interface (JNI).

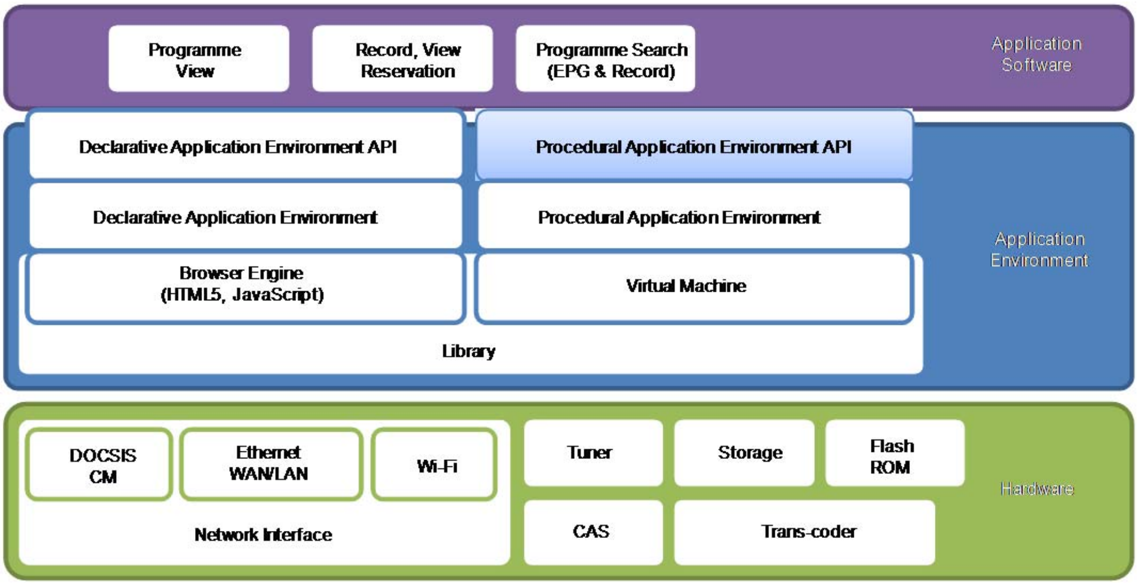

Programme View
Record, View Reservation
Programme Search (EPG & Record)
Application Software
Declarative Application Environment API
Procedural Application Environment API
Declarative Application Environment
Procedural Application Environment
Browser Engine (HTML5, JavaScript)
Virtual Machine
Library
Application Environment
DOCSIS CM
Ethernet WAN/LAN
Wi-Fi
Network Interface
Tuner
Storage
Flash ROM
CAS
Trans-coder
Hardware

Figure III.1 – Relationship of the procedural application environment and other functions in the entire functional diagram

III.1.1 API types of the procedural application environment

The following defines the API types of the procedural application environment.

- **Intent**: Intent is an inter-process communication mechanism. Through the Intent, an application can call a function of the process of another application. If necessary, the application that sends the Intent can asynchronously receive the execution result of the function.
- **ContentProvider**: ContentProvider is a mechanism to share data among plural applications. Through the ContentProvider, an application can access the database of other applications to manipulate data such as read, write, delete, search, etc.
- **Broadcast Intent**: Broadcast Intent is a mechanism to enable an application to issue the Intent to multiple processes at the same time, like broadcast but not unicast. The Broadcast Intent is primarily used for event notification throughout the system. An application that will receive the notification uses an inherited Class of Broadcast Receiver Class.
- **Class**: Class is an API that is compliant with the generic Java class definition. Unlike the Intent mechanism described above, an application uses a real entity of the Class by generating a Class instance.

An actual entity of each function can be implemented either as the native layer of C/C++ accessible through JNI (Java Native Interface), or as Java at the virtual machine layer. For any implementation scheme, an application is required to access an actual function implementation indirectly. This means that an application is prohibited to directly control, for example, hardware resources by generating a class instance. By taking such an indirect approach, an application can be separated from the internal system control functionalities, such as resource management and conflict control. This means that an application will not need to consider an actual implementation of the internal system control. In other words, the internal system control can be implemented independently from an application.

In this Recommendation, actual API implementation is required to comply with the following architecture. A typical application will be designed to access the APIs indirectly through the Intent. In this case, there will be no problem regardless of the way of implementation of the API. On the other hand, an application requiring fast response or high performance may generate the class instance to access the API. For those APIs that the application can directly access by generating the class instance, the resource control functionalities are required to be implemented outside the class itself, but to be implemented in an external entity such as a service. This means that the class will provide the API to an application to use such resources indirectly through the service. This kind of implementation can separate an upper layer application from the internal system control mechanisms.

ContentProvider database manipulation APIs are typically implemented by using the SQLite and SQLiteDatabase classes. However, other implementation ways are possible for the reason of performance optimization, etc. In this case, it is required that the same search query can be used as the SQLite.

In this Recommendation, only the public member of the class fields, constructors, and methods are defined. The definition of any private or protected member are outside the scope of this Recommendation, which can be freely implemented as necessary for the purpose of database access, Intent implementation, and so on.

This Recommendation defines the software structure of the STB and the packages that are illustrated in Figure III.2. There are three types of API packages defined in this Recommendation: (1) playback and recording APIs, (2) settings APIs, and (3) provisioning APIs. In particular, DLNA-related functions, such as content search and playback, are integrated in the (1) playback and recording APIs through which the application will be able to use the seamless content access environment, where on-air broadcasting programmes found in EPG, recorded content stored in the local storage, and content stored in the network attached storage, are seamlessly accessible from the application.

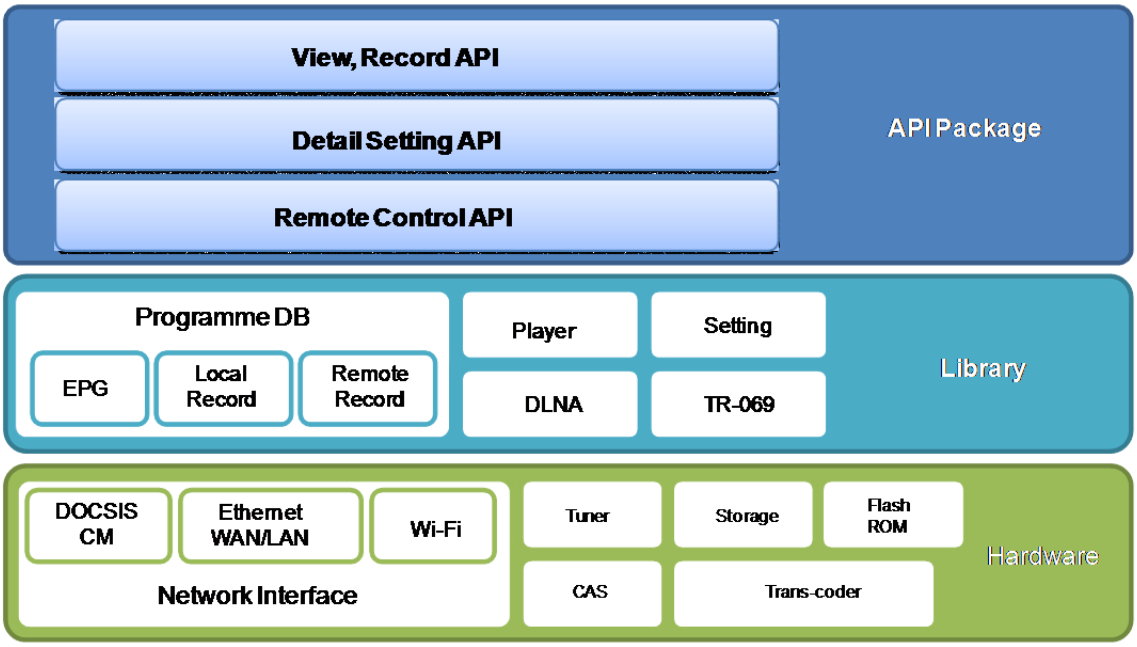
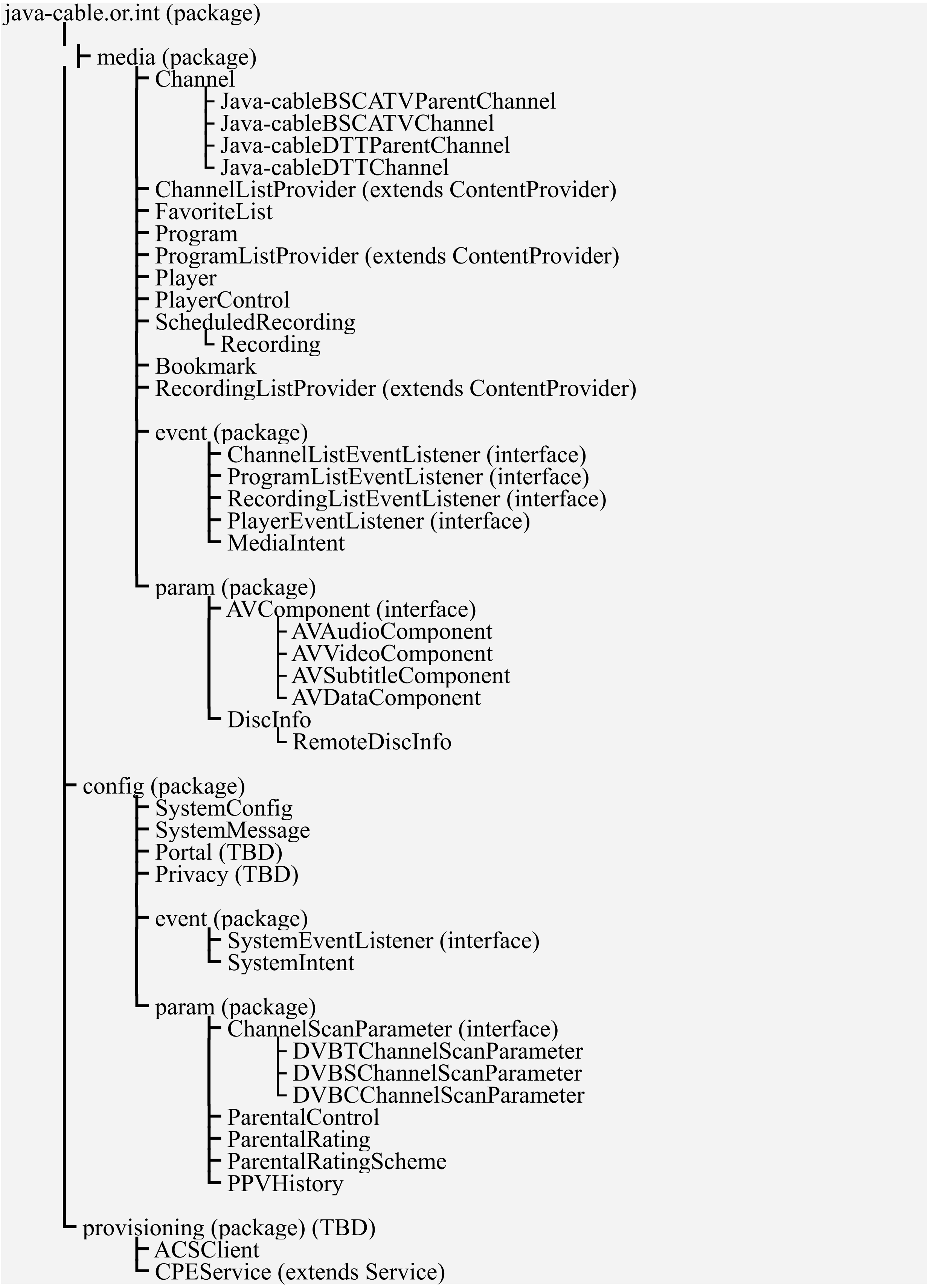


Figure III.2 – Relationship between the API package defined in this annex and other functions

III.2 Class definitions

Figure III.3 shows the class definitions defined in this Recommendation.

- All classes, except for the class noted in the following, are API classes which will be generated as an instance of the class.
- When particularly noted, the class can work with inter-process communication based on the Intent.
- In the following diagram, where the class is noted with "extends ContentProvider", indicates that the class will work as ContentProvider which is responsible for database management and data provisioning to an application.



```
java-cable.or.int (package)
  ├ media (package)
  │   ├ Channel
  │   │   ├ Java-cableBSCATVParentChannel
  │   │   ├ Java-cableBSCATVChannel
  │   │   ├ Java-cableDTTParentChannel
  │   │   └ Java-cableDTTChannel
  │   ├ ChannelListProvider (extends ContentProvider)
  │   ├ FavoriteList
  │   ├ Program
  │   ├ ProgramListProvider (extends ContentProvider)
  │   ├ Player
  │   ├ PlayerControl
  │   ├ ScheduledRecording
  │   │   └ Recording
  │   ├ Bookmark
  │   ├ RecordingListProvider (extends ContentProvider)
  │   │
  │   ├ event (package)
  │   │   ├ ChannelListEventListener (interface)
  │   │   ├ ProgramListEventListener (interface)
  │   │   ├ RecordingListEventListener (interface)
  │   │   ├ PlayerEventListener (interface)
  │   │   └ MediaIntent
  │   │
  │   └ param (package)
  │       ├ AVComponent (interface)
  │       │   ├ AVAudioComponent
  │       │   ├ AVVideoComponent
  │       │   ├ AVSubtitleComponent
  │       │   └ AVDataComponent
  │       └ DiscInfo
  │           └ RemoteDiscInfo
  │
  ├ config (package)
  │   ├ SystemConfig
  │   ├ SystemMessage
  │   ├ Portal (TBD)
  │   ├ Privacy (TBD)
  │   │
  │   ├ event (package)
  │   │   ├ SystemEventListener (interface)
  │   │   └ SystemIntent
  │   │
  │   └ param (package)
  │       ├ ChannelScanParameter (interface)
  │       │   ├ DVBTChannelScanParameter
  │       │   ├ DVBSChannelScanParameter
  │       │   └ DVBCChannelScanParameter
  │       ├ ParentalControl
  │       ├ ParentalRating
  │       ├ ParentalRatingScheme
  │       └ PPVHistory
  │
  └ provisioning (package) (TBD)
      ├ ACSClient
      └ CPEService (extends Service)
```

Figure III.3 – API package structure

It is required to implement APIs with the following requirements:

- Each method is required to return its result to the caller function with the duplicated instance of the result, not by just returning the variable itself. This approach will be able to avoid the situation where the member variable is unintentionally changed even if something is trying to directly change the variable. Each method can provide a method to change a certain member variable. In the Java language, there is no explicit destructor to delete an instance. Therefore, attention should be paid so that method implementation avoids over generation of the instance, which will cause memory shortage.
- URI for Intent or ContentProvider can be obtained from a constant of the relevant class or a method specifically designed for URI acquisition. This Recommendation does not thus define any details of the URI. There is a flexibility for implementers to define actual URIs used within the STB. Through such constants or methods, an application is able to obtain a URI to initiate the Intent.
- When an activity within an application issues the Intent with an URI, another activity already associated with the URI will be activated. The user interface of the second activity will then be shown and focused on (see the left of Figure III.4). Therefore, this Recommendation provides the following requirements to application developers when they would like to use the Intent mechanisms in the STB.
    - It is recommended that the second activity does not cover the entire user interface of the first activity. Partial transparent design of the second activity will be working well for this purpose.
    - When the process of the second activity is completed, it is recommended to immediately finish the second activity and focus again on the first activity. In this sense, it is assumed that the user interface of the second activity is a kind of simple user interface, such as a progress bar, simple message box, etc., (see Figure III.4).
    - When an activity for which the process will take longer to complete, such as a channel scan, is initiated by the Intent, the activity is required to present its progress to the user. Even when the user takes an action to move the focus to another activity, for example, by pressing the home key, it is required for the activity to continue its process in the background, although it is not focused on. On the other hand, when the user tries to cancel the activity, for example, by pressing a cancel button of the progress dialogue, the activity is required to immediately terminate itself.

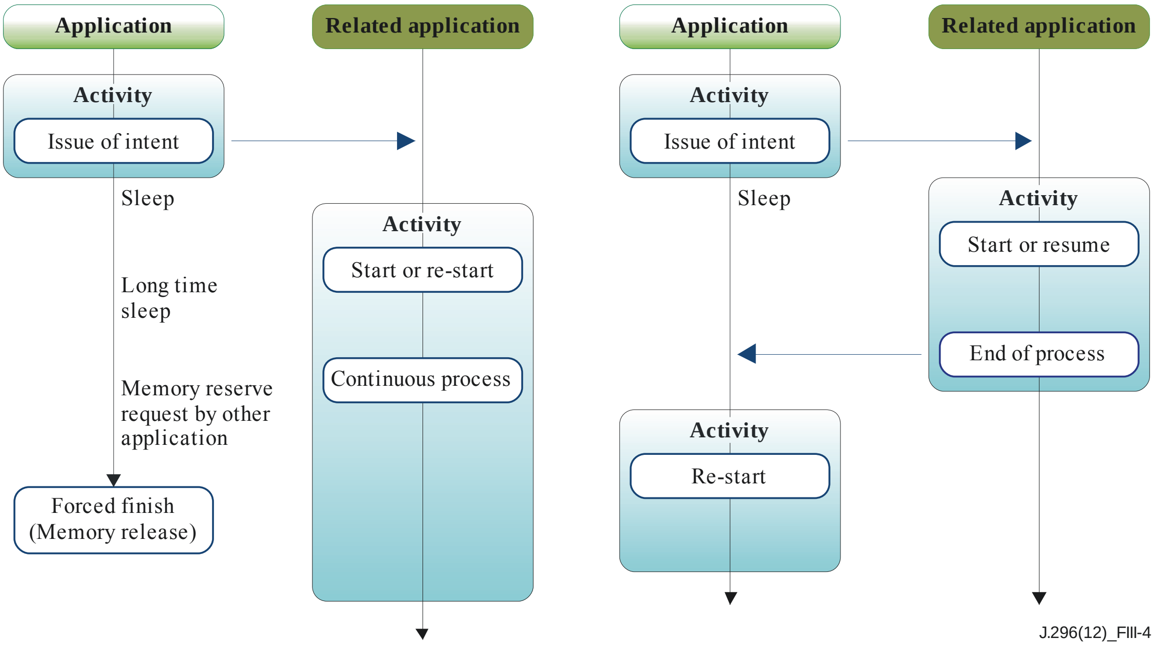


Figure III.4 – State transition diagram of the activity for Intent (left: one-way transition, right: both-way transition with suspend and resume)

III.3 API definitions

III.3.1 API package for viewing and recording

The API package is a collection of APIs for TV programme viewing and recording. The following functionalities are contained in this package:

- Player class: An API collection for the player functionalities, i.e., TV programme viewing, and content playback from local storage and remote storage through the home network (DLNA).
- Recorder class: An API collection for the recording functionalities, i.e., schedule recoding, schedule management (modify and delete), and remote schedule recoding through home network.
- Channel class and ChannelListProvider class: An API collection for channel (broadcaster) information database management, i.e., search and update of the database.
- Programme class and ProgramListProvider class: An API collection for programme information database management, i.e., search and update of the database of EPG (programmes on the air), locally recorded programmes and remotely recorded programmes.

This API package is designed so that all recorded programmes, regardless of whether locally stored or remotely stored through DLNA, can be handled with the same API, and in the same manner, from the application. In addition, it is also designed so that the same database management API can apply to the programmes contained in the EPG (programmes on the air) as well as the programmes already stored, regardless of location, e.g., local, within the home network, or in the headend (RS-DVR).

Figure III.5 illustrates the class structure defined in this API package.

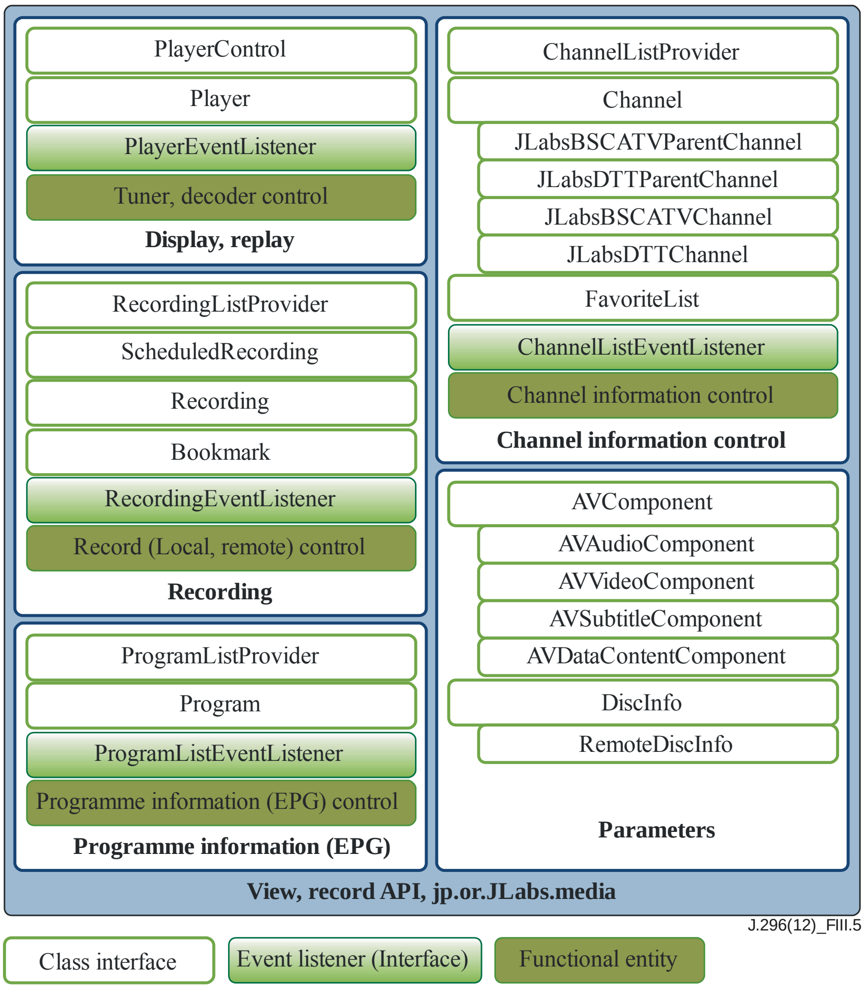


Figure III.5 – Class structure of the API package for viewing and recording

The viewing and recording functions can be utilized by the application through the player class and the recorder class through class methods or activity initiation by Intent. The database management functions can be executed to update the database, e.g., channel information update after channel scan, remote content list update of DMS, and by sending Intent to the services running on the STB.

This Recommendation defines two methods for database operations, such as search. The first method is SQL-based database search operation through ContentProvider (ChannelListProvider/ProgramListProvider/RecordingListProvider). When ContentProvider is used, ContentResolver or Cursor can be commonly used to make a search, where it is not necessary to define a specific class or method dedicated to each database.

In addition to the first method, this Recommendation defines asynchronous search APIs based on a Java function, and a callback (event listener) as a second method. This approach can be used with generic Java programing skills, and it will not require any knowledge of the ContentProvider framework. However, this method is appropriate for database search with a simplified procedure, whereas the first method is more flexible and faster because the first method is based on SQL, which is easy to make various types of database queries.

Figures III.6 and III.7 illustrate the sequence of channel information search and programme information search, respectively. The application requests updating the database of the channel or programme information through Intent in advance of the database search action. If the Intent is sent to the channel information, the Intent will initiate the channel scan. If it is sent to the programme information, the Intent will initiate to collect EPG information, the list of the content stored in the local storage and the list of the content stored in remote locations, including in-home networking or servers in the headend. If the application is equipped with a callback function for Intent, it can receive the notification of database update completion from the channel or programme information class. After these processes, the application is able to carry out actual database search through ContentProvider.

For searching of the recorded content, all the processes, including database update, are carried out within the ContentProvider framework because there is no interaction between the TV tuner and the application platform.

Figure III.8 shows the sequence of content search for the recorded content. When the application issues the request for search to the ContentProvider framework, the list of the content stored in the local storage will be returned to the application in accordance with the query specified by the application.

Figure III.9 shows the sequence of content search for the content stored in the DMS connected to the home network. The application will firstly obtain the list of DMSs within the home network through the API of the class method. Then, the application will choose a DMS from the list, for example, by user's operation to the application. For the specified DMS, the application should obtain the ContentProvider URI through the class method. With the obtained URI, the application will carry out content search through the ContentProvider framework.

The timing of updating the content list database is implementation dependent of the APIs in this Recommendation, i.e., the list can be updated at every moment when a query is issued, or it can be also updated at intervals, etc. The algorithm of update control can be optimized at the implementation layer of the APIs.

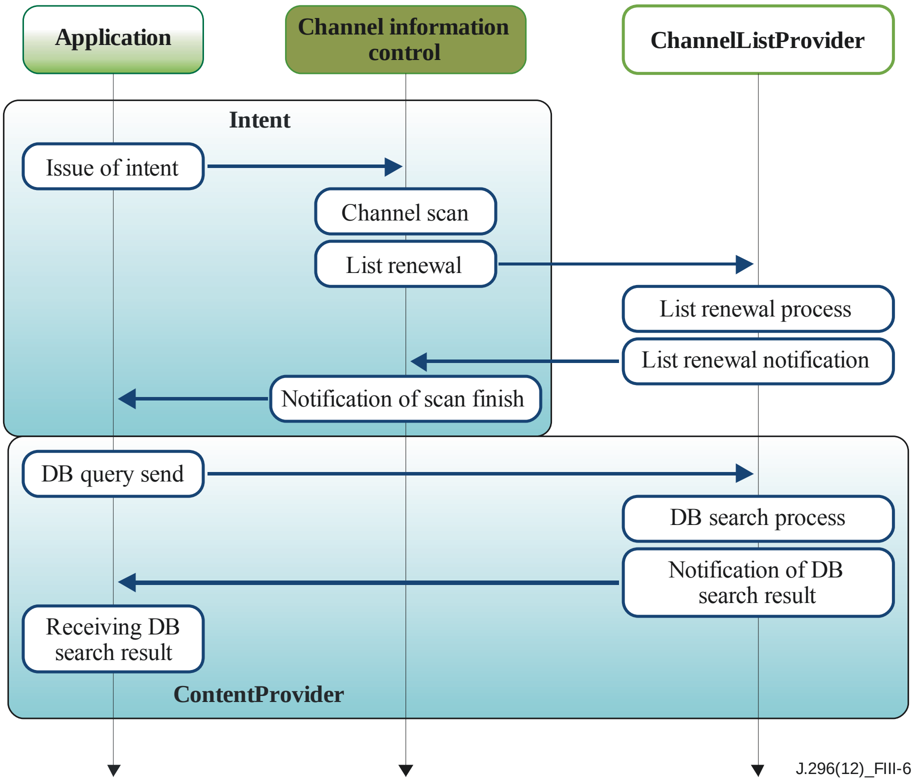


Figure III.6 – Sequence of channel information search

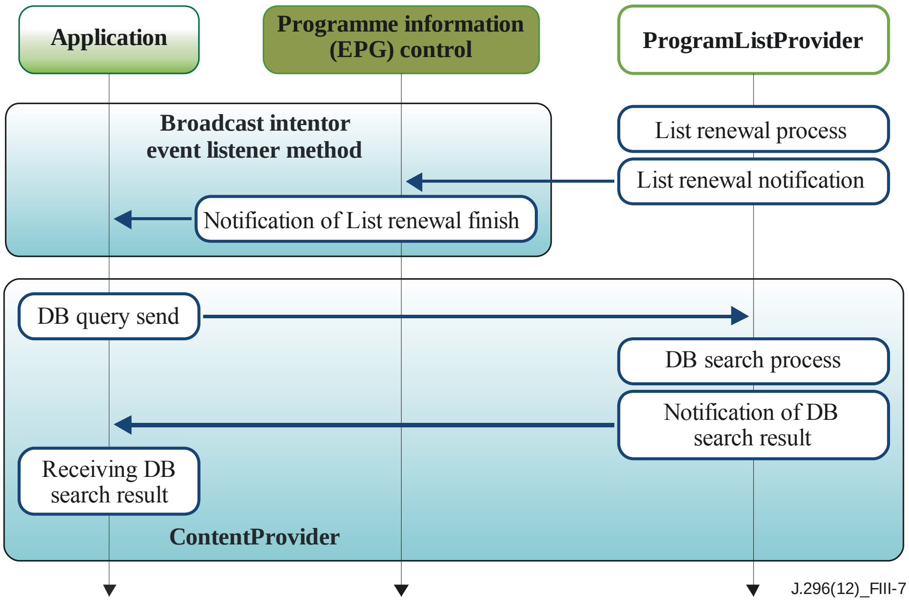

Application
Programme information (EPG) control
ProgramListProvider
Broadcast intentor event listener method
List renewal process
List renewal notification
Notification of List renewal finish
DB query send
DB search process
Notification of DB search result
Receiving DB search result
ContentProvider
J.296(12)_FIII-7

Figure III.7 – Sequence of TV programme information search

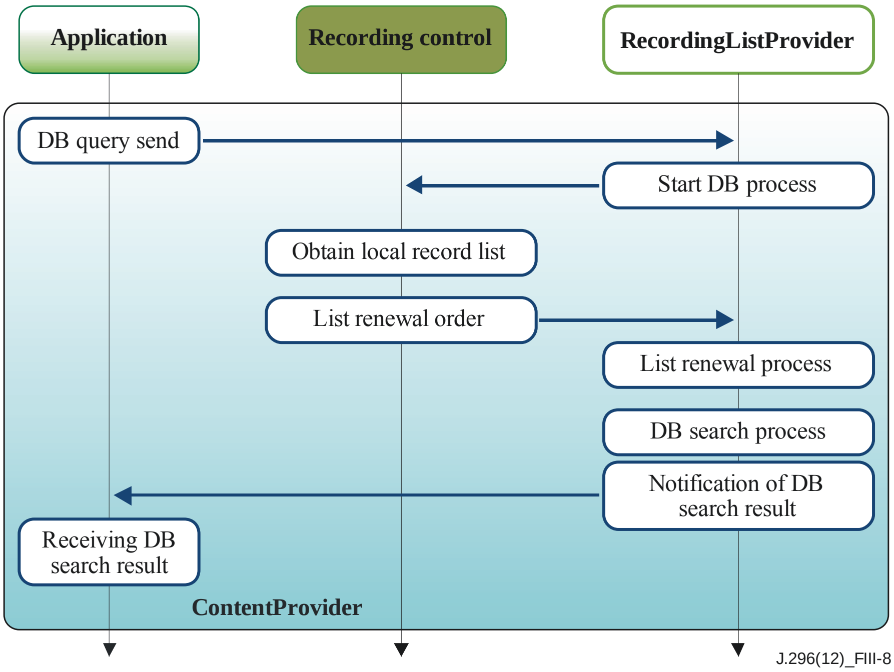


Figure III.8 – Sequence of content search for the content stored in the local storage

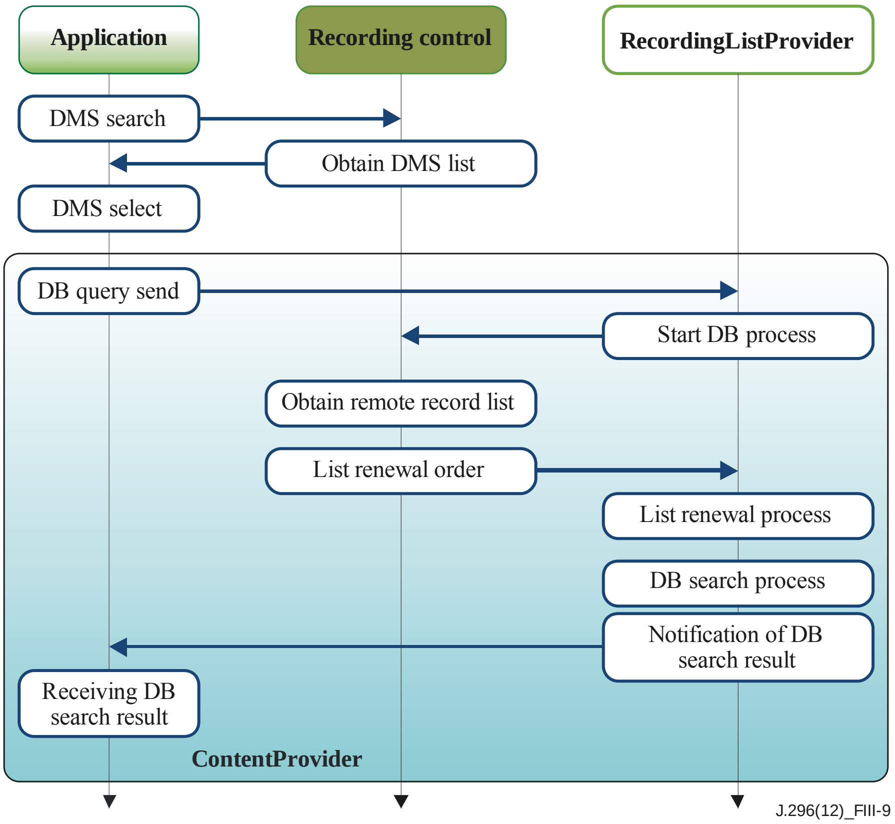


Figure III.9 – Sequence of content search for the content stored in the DMS connected to the home network

III.3.2 Channel class

The hierarchy of the class is described as follows:

java.lang.Object

└**java-cable.or.int.media.Channel**

The same notation will apply to all the remaining class definitions in this Recommendation.

The channel class is responsible for information and control of the channels that can be received by the STB. The channel class also provides the URI for Intent or ContentProvider to obtain the channel information or to carry out a channel scan.

III.3.2.1 Fields

Identifiers and type definitions related to this class are defined in the following tables (Tables III.1 to III.4).

Table III.1 – Channel type identifiers

Name	Value	Description
ID_ANALOG	0	Refers to conventional analogue broadcasting channels.
ID_DVB_C	10	Refers to DVB-C channels.
ID_DVB_S	11	Refers to DVB-S channels.
ID_DVB_T	12	Refers to DVB-T channels.
ID_DVB_SI_DIRECT	13	Refers to DVB-SI channels.
ID_DVB_C2	14	Refers to DVB-C2 or DVB-C channels.
ID_DVB_S2	15	Refers to DVB-S2 or DVB-S channels.
ID_DVB_T2	16	Refers to DVB-T2 or DVB-T channels.
ID_ISDB_C	20	Refers to ISDB-C channels.
ID_ISDB_S	21	Refers to ISDB-S channels.
ID_ISDB_T	22	Refers to ISDB-T channels.
ID_ISDB_C_SI	23	Refers to SI channel of ISDB-C.
ID_ATSC_T	30	Refers to ATSC channels.
ID_IPTV_SDS	40	Refers to SD&S channel of the Open IPTV Forum specification-based system.
ID_IPTV_URI	41	Refers to URI (DVB MCAST URI) of the Open IPTV Forum specification-based system.

Table III.2 – Broadcast network identifiers

Name	Value	Description
JAVA-CABLE_DTT_NETWORK	0	Refers to digital terrestrial television network.
JAVA-CABLE_BS_NETWORK	1	Refers to digital satellite television network.
JAVA-CABLE_CATV_NETWORK	2	Refers to digital cable television network.
JAVA-CABLE_ALL_NETWORK	3	Refers to all of the above-described networks.

Table III.3 – URIs

Name	Description
CONTENT_URI	The ContentProvider URI for channel list search of the ChannelListProvider class.
CONTENT_FAVORITE_URI	The ContentProvider URI for favourite channel list search.
CHANNEL_SCAN_URI	The Intent URI to initiate the channel scan process.

The following table defines the parameters that can be specified in Intent with putExtra or getExtra.

Table III.4 – Intent parameters

Name	Type	Description
SCAN_NETWORK_TYPE	Integer	The broadcasting network category listed in Table III.2 to be scanned by Intent for channel scan with CHANNEL_SCAN_URI. The default value is JAVA-CABLE_ALL_NETWORK.
ID	Integer	The identifier of the player to carry out channel change (the same as PlayerID).

III.3.2.2 Constructors

Constructors related to this class are defined in Table III.5.

Table III.5 – Constructors of the channel class

Channel()	
Parameter	None
Description	It creates and initializes an instance of the channel class.
Channel(int idType)	
Parameter	idType: The identifier of the channel type listed in Table III.1.
Description	It creates and initializes an instance of the channel class with the specific channel type corresponding to the parameter idType.
Channel(int idType, int networkId, int tsId, int serviceId)	
Parameter	idType: The identifier of the channel listed in Table III.1. networkId: Network ID tsId: Transport stream ID serviceId: Service ID
Description	It creates and initializes an instance of the channel class with the specific channel information specified by idType, networkId, tsId and serviceId.

III.3.2.3 Methods

Methods related to this class are defined in Table III.6.

Table III.6 – Methods of the channel class

Parameter	None
Return Value	The identifier of the channel type listed in Table III.1.
Description	It retrieves the identifier of the channel type.
java.lang.String **getName()**	
Parameter	None
Return Value	A string of the channel name to be retrieved.
Description	It retrieves the channel name as a string. It will be typically a three letter name, e.g., NHK, CNN, etc.
void **setName**(java.lang.String name)	
Parameter	name: A string of the channel name to be set.
Return Value	None
Description	It specifies the channel name with the string specified by the "name".

Table III.6 – Methods of the channel class

java.lang.String **getLongName()**	
Parameter	None
Return Value	A string of the long channel name to be retrieved.
Description	It retrieves the long channel name as a string. getLongName() will return a Longer and more detailed string than getName().
void **setLongName**(java.lang.String name)	
Parameter	name: A string of the channel name to be set.
Return Value	None
Description	It specifies the long channel name by a string.
java.lang.String **getDescription**()	
Parameter	None
Return Value	A string of the description of the channel.
Description	It retrieves the description of the channel as a string.
void **setDescription**(java.lang.String desc)	
Parameter	desc: A string of the description of the channel.
Return Value	None
Description	It specifies the description of the channel with the string specified by the "desc".
int **getTunerId**()	
Parameter	None
Return Value	An integer value of the tuner ID.
Description	It retrieves the tuner ID associated with the channel class.
int **getNetworkId**()	
Parameter	None
Return Value	An integer value of the network ID.
Description	It retrieves the network ID of the channel currently specified by the class instance.
int **getTransportStreamId**()	
Parameter	None
Return Value	An integer value of the transport stream ID.
Description	It retrieves the transport stream ID of the channel currently specified by the class instance.
int **getServiceId**()	
Parameter	None
Return Value	An integer value of the service ID.
Description	It retrieves the service ID of the channel currently specified by the class instance.
boolean **isFavorite**()	
Parameter	None
Return Value	True if there is a favourite channel list that contains the channel currently specified by the class instance.
Description	It returns whether or not there is a favourite channel list that contains the channel currently specified by the class instance.
java.lang.String[] **getFavIDs**()	

Table III.6 – Methods of the channel class

Parameter	None
Return Value	An array of a string of the identifier of all the favourite channel lists that contain the channel currently specified by the class instance.
Description	It returns all the favourite channel lists that contain the channel currently specified by the class instance.
boolean **isLocked**()	
Parameter	None
Return Value	True if the channel currently specified by the class instance is locked by parental control.
Description	It returns whether or not the channel currently specified by the class instance is locked by parental control.
boolean **isManualBlocked**()	
Parameter	None
Return Value	True if the channel currently specified by the class instance is manually locked by a PIN code.
Description	It returns whether or not the channel currently specified by the class instance is manually locked by a PIN code.
boolean **isAuthorized**()	
Parameter	None
Return Value	True if the channel currently specified by the class instance is authorized to be seen by the conditional access system.
Description	It returns whether or not the channel currently specified by the class instance is authorized to be seen by the conditional access system equipped with the STB.
java.lang.String[] **getGenre**()	
Parameter	None
Return Value	An array of a string of the genre of the channel currently specified by the class instance.
Description	It retrieves an array of a string of the genre of the channel currently specified by the class instance.
void **setGenre**(java.lang.String[] genre)	
Parameter	genre: An array of a string of the genre of the channel currently specified by the class instance.
Return Value	None
Description	It specifies the genre(s) of the channel currently specified by the class instance. Multiple genres can be specified by an array of a string.
boolean **isHidden**()	
Parameter	None
Return Value	True if the channel is specified as a channel to be concealed on the EPG.
Description	It returns whether or not the channel, which is specified by the class instance, is specified as a channel to be concealed on the EPG.
void **setHidden**(boolean flag)	
Parameter	flag: "true" to make the channel be concealed on the EPG, otherwise "false".
Return Value	None

Table III.6 – Methods of the channel class

Description	It specifies whether or not the channel, which is specified by the class instance, should be concealed on the EPG.
java.lang.String **getLogoFilePath**()	
Parameter	None
Return Value	A string of the path of the logo file. Null if there is no logo file obtained in the STB.
Description	It retrieves the file path of the logo image file associated with the channel specified by the class instance.
void **setLogoFilePath**(java.lang.String path)	
Parameter	path: A string of the path of the logo file.
Return Value	None
Description	It associates an image file locally stored in the STB with the channel specified by the class instance as a logo.
java.lang.String **getLogoURL**()	
Parameter	None
Return Value	A string of the URL of the logo file available in the network. Null if there is no logo file available in the network.
Description	It retrieves the URL of the logo image file associated with the channel specified by the class instance if it is available in the network via HTTP or HTTPS.
void **setLogoURL**(java.lang.String url)	
Parameter	url: A string of the URL of the logo file available in the network.
Return Value	None
Description	It associates an image file available in the network via HTTP or HTTPS with the channel specified by the class instance.
java.lang.String **getField**(java.lang.String[] fieldId)	
Parameter	fieldId: A string of the identifier of the metadata to be retrieved.
Return Value	The content of the metadata specified with the parameter.
Description	It retrieves the content of the metadata associated with the channel specified by the class instance. This method can be used to retrieve the values of the metadata that cannot be obtained with other method of the class. This Recommendation only defines the method API, but the format of the metadata and the operation of the metadata are outside the scope of this Recommendation.
int **getServiceType**()	
Parameter	None
Return Value	An integer value indicating the service type of the channel.
Description	It retrieves the service type of the channel specified by the class instance. The service type value is defined by the broadcasting system standard used in each region.
int **getRemoteControlKeyNumber**()	
Parameter	None
Return Value	An integer value of the short cut numeric button of the remote controller assigned to the channel. Zero (0) if there is no assignment.
Description	It retrieves the short cut numeric button of the remote controller assigned to the channel specified by the class instance.

Table III.6 – Methods of the channel class

boolean **setRemoteControlKeyNumber**(int key)	
Parameter	key: An integer value of the short cut numeric button of the remote controller to be assigned to the channel. Specifying zero (0) will remove the short cut association.
Return Value	True if successful. False if not.
Description	It assigns a numeric button of the remote controller to the channel specified by the class instance.
int **getThreeDigitNumber**()	
Parameter	None
Return Value	An integer value of the three-digit channel number associated with the channel.
Description	It retrieves the three-digit channel number associated with the channel specified by the class instance.
android.net.Uri **getChannelUri**()	
Parameter	None
Return Value	The Intent URI for channel change.
Description	It retrieves the Intent URI to be used for channel change to the channel corresponding to the information associated with the class instance.

III.3.3 Java-cableBSCATVParentChannelClass

java.lang.Object

└java-cable.or.int.media.Channel

  └**java-cable.or.int.media.Java-cableBSCATVParentChannel**

The Java-cableBSCATVParentChannelClass holds a collection of the information of all the channels having the same network id and the information commonly applicable to all the channels, particularly for satellite or cable television broadcasting including retransmission of the satellite television channels over cable network.

III.3.3.1 Methods

Methods related to this class are defined in Table III.7.

Table III.7 – Methods of the Java-cableBSCATVParentChannel class

Channel[] **getChildChannel**()	
Parameter	None
Return Value	An array of the channel class constituting the same satellite or cable television network.
Description	It retrieves an array of the channel class instances corresponding to the channels constituting the same satellite or cable television network.
java.lang.String **getSignalLevel**()	
Parameter	None
Return Value	A string indicating the signal level of the channel.
Description	It retrieves the signal level of the channel specified by the class instance.
int **getNetworkType**()	
Parameter	None
Return Value	An integer value indicating the broadcast network identifier.

Table III.7 – Methods of the Java-cableBSCATVParentChannel class

Description	It retrieves the broadcast network identifier of the channel listed in Table III.2 specified by the class instance.
java.util.Date **getLastUpdateDate**()	
Parameter	None
Return Value	A Date object indicating the last modified date and time of the channel information.
Description	It retrieves the last modified date and time of the channel information obtained by the channel scan procedure.

III.3.4 Java-cableBSCATVChannel Class

java.lang.Object

└java-cable.or.int.media.Channel

└java-cable.or.int.media.Java-cableBSCATVChannel

The Java-cableBSCATVChannel class is responsible for holding the channel-specific information of satellite and cable television broadcasting.

III.3.4.1 Methods

Methods related to this class are defined in Table III.8.

Table III.8 – Methods of the Java-cableBSCATVChannel class

JLabsBSCATVParentChannel **getParentChannel**()	
Parameter	None
Return Value	The instance of the Java-cableBSCATVParentChannel class related to the channel specified by the class instance of the Java-cableBSCATVChannel class.
Description	It retrieves the class instance of Java-cableBSCATVParentChannel related to the channel specified by the class instance of the Java-cableBSCATVChannel class.

III.3.5 Java-cableDTTParentChannel Class

java.lang.Object

└java-cable.or.int.media.Channel

└java-cable.or.int.media.Java-cableDTTParentChannel

The Java-cableDTTParentChannelClass holds a collection of the information of all the channels having the same network id and the information commonly applicable to all the channels, particularly for digital terrestrial television broadcasting including retransmission of the digital terrestrial television channels over cable network.

III.3.5.1 Methods

Methods related to this class are defined in Table III.9.

Table III.9 – Methods of the Java-cableDTTParentChannel class

Channel[] **getChildChannel**()	
Parameter	None
Return Value	An array of the channel classes constituting the same digital terrestrial television network.
Description	It retrieves an array of the channel class instances corresponding to the channels constituting the same digital terrestrial television network.
int **getAreaCode**()	
Parameter	None
Return Value	An integer value indicating the area code of the channels.
Description	It retrieves the area code of the channels specified by the class instance. The value of the area code is defined by the broadcasting standard used in each region.
int **getBranchNumber**()	
Parameter	None
Return Value	An integer value indicating the branch number to be appended to the three-digit channel number.
Description	It retrieves the branch number of the channels if it is available.
Channel[] **getBranchChannel**()	
Parameter	None
Return Value	An array of the channel classes associated with the same network id.
Description	It retrieves
java.lang.String **getSignalLevel**()	
Refer to III.3.3.1	
int **getNetworkType**()	
Refer to III.3.3.1	
java.util.Date **getLastUpdateDate**()	
Refer to III.3.3.1	

III.3.6 Java-cableDTTChannel Class

java.lang.Object

└java-cable.or.int.media.Channel

└**java-cable.or.int.media.Java-cableDTTChannel**

The Java-cableDTTChannel class is responsible for holding the channel-specific information of digital terrestrial television broadcasting.

III.3.6.1 Methods

Methods related to this class are defined in Table III.10.

Table III.10 – Methods of the Java-cableDTTChannel class

Java-cableDTTParentChannel **getParentChannel**()	
Parameter	None
Return Value	The instance of the Java-cableDTTParentChannel class related to the channel specified by the class instance of the Java-cableDTTChannel class.
Description	It retrieves the class instance of Java-cableDTTParentChannel related to the channel specified by the class instance of the Java-cableDTTChannel class.

III.3.7 FavoriteList Class

java.lang.Object

└java-cable.or.int.media.FavoriteList

The FavoriteList class is responsible for favourite channel handling of the STB. It includes a channel collection to be used for up-down zapping operation, a list of the favourite channels specified by the user, etc.

III.3.7.1 Fields

Fields related to this class are defined in Table III.11.

Table III.11 – URIs

Name	Description
CONTENT_URI	The ContentProvider URI to be used for favourite channel list search. It is the same as Channel.CONTENT_FAVORITE_URI.

III.3.7.2 Constructors

Constructors related to this class are defined in Table III.12.

Table III.12 – Constructors of the FavoriteList class

FavoriteList()	
Parameter	None
Description	It creates and initializes an instance of the FavoriteList class.
FavoriteList(java.lang.String name)	
Parameter	name: The name of the favourite list.
Description	It creates and initializes an instance of the FavoriteList class with the specific name of the list.
FavoriteList(java.lang.String name, Channel[] channels)	
Parameter	name: The name of the favourite list. channels: An array of the channels to be contained in the favourite list.
Description	It creates and initializes an instance of the FavoriteList class with the specific name of the list and the specific channels to be contained in the list.

III.3.7.3 Methods

Methods related to this class are defined in Table III.13.

Table III.13 – Methods of the FavoriteList class

java.lang.String **getFavId**()	
Parameter	None
Return Value	A string indicating the identifier of the favourite list.
Description	It retrieves the identifier of the favourite list. The identifier is a unique value to specify the particular favourite list within the STB.
java.lang.String **getName**()	
Parameter	None
Return Value	A string indicating the name of the favourite list.
Description	It retrieves the name of the favourite list.
void **setName**(java.lang.String name)	
Parameter	name: The name of the favourite list.
Return Value	None
Description	It specifies the name of the favourite list. The name will be provided by the user.
Channel[] **getChannelList**()	
Parameter	None
Return Value	An array of the channel classes contained in the favourite list.
Description	It retrieves the channel classes contained in the favourite list corresponding to the Favourite class instance.
void **setChannelList**(Channel[] channels)	
Parameter	channels: An array of the channel classes to be registered to the favourite list.
Return Value	None
Description	It registers the channels to the favourite list.
Channel[] **getChannelByName**(java.lang.String name)	
Parameter	name: The name of the favourite list.
Return Value	An array of the channel classes corresponding to the channels contained in the specified favourite list.
Description	It retrieves the array of the channel classes corresponding to the channels contained in the specified favourite list.
Channel **getChannelByTriplet**(int networkId, int tsId, int serviceId)	
Parameter	networkId: Network ID tsId: Transport stream ID serviceId: Service ID
Return Value	The channel class instance of the channel corresponding to the specified triplet.
Description	It retrieves the channel class corresponding to the channel contained in the favourite list associated with the specified triplet, i.e., network ID, transport stream ID and service ID.

III.3.8 ChannelListProvider Class

java.lang.Object

└android.content.ContentProvider

└**java-cable.or.int.media.ChannelListProvider**

The ChannelListProviderClass is responsible for database management of the channel list. It provides search, update, etc.

III.3.8.1 Column Labels

Table III.14 defines the column labels to be used for database search with the ContentProvider framework.

Table III.14 – Database labels for channel list

Label name (Channel field)	Type	Value or corresponding class method
ID	Integer	Identifier of each element (rows)
CHANNEL_URI	String	Channel#getChannelUri()
CALLSIGN	String	Channel#getName()*
NAME	String	Channel#getName()
CHANNEL_NAME	String	Channel#getLongName()*
LONG_NAME	String	Channel#getLongName()
CHANNEL_NUMBER	String	Channel#getThreeDigitNumber()*
THREE_DIGIT	String	Channel#getThreeDigitNumber()
KEY_NUMBER	Integer	Channel#getRemoteControlKeyNumber()
DESCRIPTION	String	Channel#getDescription()
CHANNEL_TYPE	Integer	Channel#getChannelType()
TUNER_ID	Integer	Channel#getTunerId()
NETWORK_ID	Integer	Channel#getNetworkId()
TS_ID	Integer	Channel#getTransportStreamId()
SERVICE_ID	Integer	Channel#getServiceId()
FAVORITE	Integer	Channel#isFavorite()
FAV_IDS	String	Channel#getFavIDs()
LOCKED	Integer	Channel#isLocked()
MANUAL_BLOCKED	Integer	Channel#isManualBlocked()
AUTHORIZED	Integer	Channel#isAuthorized()
GENRE	String	Channel#getGenre()
HIDDEN	Integer	Channel#isHidden()
LOGO_FILE_PATH	String	Channel#getLogoFilePath()
LOGO_URL	String	Channel#getLogoURL()
SERVICE_TYPE	Integer	Channel#getServiceType()
NETWORK_TYPE	Integer	Java-cableBSCATVParentChannel#getNetworkType() Java-cableDTTParentChannel#getNetworkType()
LAST_UPDATE	Integer	Java-cableBSCATVParentChannel#getLastUpdateDate() Java-cableDTTParentChannel#getLastUpdateDate ()

Table III.14 – Database labels for channel list

BRANCH_NUMBER	Integer	Java-cableDTTChannel#getBranchNumber()
AREA_CODE	Integer	Java-cableDTTParentChannel#getAreaCode()
* These labels are defined aiming to maintain the compatibility with other existing APIs. NOTE 1 – Labels are referred to as the **Channel.[Label name]** format. NOTE 2 – In this table, each parameter originally defined as Boolean is defined as Integer. In this case, value 0 means "true" and value 1 means "false". NOTE 3 – In this table, each parameter originally defined as Array of characters is defined as String. In this case, original characters are connected by a comma and form one String. NOTE 4 – In this table, the parameter originally defined as java.util.Data is defined as Integer. In this case, "UNIX time" (the elapsed time from 00:00:00 on 1st January 1970) shall be used.		

Table III.15 defines the column labels to be used for database search with the ContentProvider framework.

Table III.15 – Database labels for favourite list

Label name (FavoriteList field)	Type	Values or corresponding class method
ID	Integer	Identifier of each element (rows)
FAV_NAME	String	FavoriteList#getName()
FAV_ID	String	FavoriteList#getFavId()
CHANNELS	String	FavoriteList#getChannelList() Channel.ID value of ContentProvider of channel information shall be described as Comma separated values.
NOTE – Labels are referred as the **FavoriteList.[Label name]** format.		

III.3.8.2 Method

Methods related to this class are defined in Table III.16.

In this Recommendation, explanations of common methods of the ContentProvider class (getType, query, insert, delete, and update) are skipped.

Table III.16 – Methods of ChannelListProviderClass

Channel[] **getChannelByName**(java.lang.String name)	
Argument	name: Search query string
Return Value	Arrays of all channel information whose channel name was matched (partially or fully) with the search query.
Description	Run the partial match retrieval of channel name and get channel information.
Channel **getChannelByTriplet**(int networkId, int tsId, int serviceId)	
Argument	networkId: NetworkID tsId: TSID serviceId: ServiceID
Return Value	Channel Information corresponding to the above three IDs.
Description	Get the channel information identified by NetworkID, TSID, and ServiceID.

Table III.16 – Methods of ChannelListProviderClass

FavoriteList **getFavoriteList**(String favId)	
Argument	favId: Favourite List ID
Return Value	Favourite List corresponding to the above ID.
Description	Get the favoriteList associated with the Favourite List ID.
FavoriteList **getCurrentFavoriteList**()	
Argument	None
Return Value	Active Favourite List
Description	Get the favourite list that is currently used. (In this case, only the channels described in the favourite list are shown by the Up/Down key operation of a remote.)
boolean **activateFavoriteList**(String favId)	
Argument	favId: Favourite List ID
Return Value	Case of successful completion: true
Description	Get the favourite list specified by Favourite List ID. (When any favourite list is active, only channels described in that list are shown by the Up/Down key operation of a remote.)
FavoriteList[] **getFavoriteLists**()	
Argument	Note
Return Value	All favourite lists
Description	Get all favourite lists as an array.
boolean **commitFavoriteLists**(FavoriteList[] favLists)	
Argument	favLists: Array of favourite lists
Return Value	Case of successful completion: true
Description	Adopt the favourite list specified by the argument.
void **startScan**(channelScanParameter param, boolean replaceExisting)	
Argument	param: Parameter for channel scan replaceExisting: True: Allow the override of existing channel settings. False: Only allow the registration of additional channels.
Return Value	None
Description	Start the channel scan process.
void **stopScan**()	
Argument	None
Return Value	None
Description	Stop currently-running channel scan process.
void **addChannel**(Channel ch)	
Argument	ch: Channel information of additional channel

Table III.16 – Methods of ChannelListProviderClass

Return Value	None
Description	Register target channel.
void **deleteChannel**(Channel ch)	
Argument	ch: Channel information of unregistered channel
Return Value	None
Description	Unregister target channel.
void **setChannelListEventListener**(ChannelListEventListener listenerClass)	
Argument	listenerClass: Class for the event listener
Return Value	None
Description	Set the class specified by the argument as the event listener. This event listener will be called when any event related to the channel information is invoked.
boolean **isJava-cableChannelTree**()	
Argument	None
Return Value	If the channel information is stored as the channel tree: true
Description	Confirm the stored format of the search result of channels. When it returns true, channel information is stored in the tree format. When it returns false, channel information is stored as the list of channel number and channel sub-number, which is used by the user.
void **setJava-cableChannelTree**(boolean flag)	
Argument	flag: Set true to store the channel information in tree format.
Return Value	None
Description	Set the stored format of the search result of channels. When it returns true, channel information is stored in the tree format. When it returns false, channel information is stored as the list of channel number and channel sub-number, which is used by the user.
int **getLastChannelNetworkType**()	
Argument	None
Return Value	Broadcasting network ID of last channel. (Table III.2)
Description	Get the broadcasting network ID of last channel.
int **getLastChannelThreedigit**()	
Argument	None
Return Value	Logical channel number of last channel.
Description	Get the logical channel number of last channel.
boolean **isSeamlessChannelListing**()	
Argument	None
Return Value	True: Up/Down channel change is continued over the different broadcasting networks (digital terrestrial television, digital satellite television, and digital cable television). False: Up/Down channel change is working in the currently selected broadcasting network.

Table III.16 – Methods of ChannelListProviderClass

Description	Confirm selection channel pattern by channel Up/Down key of remote controller
void **setSeamlessChannelListing**(boolean flag)	
Argument	True: Up/Down channel change is continued over the different broadcasting networks (digital terrestrial television, digital satellite television, and digital cable television). False: Up/Down channel change is working in the currently selected broadcasting network.
Return Value	None
Description	Set the channel change behaviour of Up/Down key.
void **changeNetwork**(int networkType)	
Argument	networkType: Active broadcasting network IDs (Table III.2)
Return Value	None
Description	Set the broadcasting network IDs which are active for channel selection.

III.3.9 ProgramClass

java.lang.Object

└java-cable.or.int.media.Program

This class provides the management function of EPG information stored in STB. Also, this class provides URIs for applications. Each application can get the detailed programme information through them.

III.3.9.1 Field

Identifiers and type definitions related to this class are defined in Tables III.17 and III.18.

Table III.17 – Programme type identifier

Field name	Value	Description
ID_TVA_CRID	0	This field indicates TV-Antime CRID is used as the programme information. NOTE – This field is defined aiming to keep compatibility with Open IPTV Forum DAE[].
ID_DVB_EVENT	1	This field indicates DVB programme information is used as the programme information.
ID_TVA_GROUP_ID	2	This field indicates TV-Antime group CRID is used as the programme information.

Table III.18 – URI Field

Field name	Description
CONTENT_URI	This field indicates ContentProviderURI of ProgramListProvider.

III.3.9.2 Method

Methods related to this class are defined in Table III.19.

Table III.19 – Method of programme class

java.lang.String **getName**()	
Argument	None.
Return Value	String indicating the name of the programme.
Description	Get the name of the programme as string format.
void **setName**(java.lang.String name)	
Argument	name: Programme name.
Return Value	None.
Description	Set the specified Programme name string as the programme title.
java.lang.String **getDescription**()	
Argument	None.
Return Value	String indicating the description of the programme.
Description	Get the description of the programme as string format.
void **setDescription**(java.lang.String desc)	
Argument	desc: Description of the programme.
Return Value	None.
Description	Set the specified value as the description of the programme.
java.lang.String **getLongDescription**()	
Argument	None.
Return Value	String indicating the detailed description of programme.
Description	Get the detailed description of the programme as string format.
void **setLongDescription**(java.lang.String desc)	
Argument	desc: Detailed description of programme.
Return Value	None.
Description	Set the specified value as the detailed description of the programme.
java.util.Date **getStartTime**()	
Argument	None
Return Value	Start time of the programme.
Description	Get the start time of programme from the programme information.
void **setStartTime**(java.util.Date date)	
Argument	date: Start time of the programme.
Return Value	None.
Description	Set the specified value as the start time of the programme.
int **getDuration**()	
Argument	None

Table III.19 – Method of programme class

Return Value	Length of the programme (scale:sec).
Description	Get the programme length from the programme information.
void **setDuration**(int dur)	
Argument	dur: Length of the programme (scale: sec).
Return Value	None
Description	Set the specified value as the length of the programme.
int **getProgramID**()	
Argument	None.
Return Value	Programme identifier .
Description	Get the programme identifier. Programme identifier is the unique ID to specify the programme information in the DB. In Annex X system, event_id contained in EIT shall be used as defined in [ARIB TR-B14].
void **setProgramID**(int id)	
Argument	id: Programme identifier.
Return Value	None.
Description	Set the specified value as the programme identifier.
int **getProgramIDType**()	
Argument	None.
Return Value	Programme type identifier.
Description	Get the programme type identifier (defined in Table III.17) of the programme information.
void **setProgramIDType**(int idType)	
Argument	idType: Programme type identifier.
Return Value	None.
Description	Set the programme type identifier of the programme.
Channel **getChannel**()	
Argument	None.
Return Value	Channel information of the programme.
Description	Get the channel information that the programme will be broadcast.
int **getEpisode**()	
Argument	None.
Return Value	Episode number of the programme.
Description	Get the episode number of the programme. In Annex X system, episode_number contained in EIT shall be used as defined in [ARIB TR-B14].

Table III.19 – Method of programme class

void **setEpisode**(int episode)	
Argument	episode: Episode number.
Return Value	None.
Description	Set the episode number of the programme.
int **getTotalEpisodes**()	
Argument	None.
Return Value	Total number of episodes.
Description	Get the total number of episodes in which the programme is included.
void **setTotalEpisodes**(int episodes)	
Argument	episode: Total number of episodes.
Return Value	None.
Description	Set the total number of episodes of the programme.
ParentalRating[] **getParentalRating**()	
Argument	None.
Return Value	Parental lock rating information of the programme.
Description	Get parental lock rating information of the programme.
void **addParentalRating**(ParentalRating rating)	
Argument	rating: Parental lock rating information.
Return Value	None.
Description	Set parental lock rating information to the programme.
boolean **isLocked**()	
Argument	None.
Return Value	True: If the programme is under any parental control.
Description	Return true if the watching of target programme is restricted by any parental control.
boolean **isBlocked**()	
Argument	None.
Return Value	True: If the watching of target programme is restricted by any PIN lock.
Description	Return true if the watching of the programme is restricted by any PIN lock.
boolean **hasSubtitles**()	
Argument	None.
Return Value	True: If the programme has closed caption.
Description	Confirm if the programme has closed caption.
void **setSubtitles**(boolean flag)	
Argument	flag: Set true to indicate the programme has closed caption

Table III.19 – Method of programme class

Return Value	None.
Description	Manually set the flag which indicating whether the programme has closed caption or not.
boolean **isHD**()	
Argument	None.
Return Value	True: If the programme is HD.
Description	Confirm whether the programme resolution is HD (720p or higher).
void **setHD**(boolean flag)	
Argument	Flag: Set true to indicate the programme is HD.
Return Value	Note.
Description	Manually set the flag which indicating whether the programme is HD (720p or higher) or not.
boolean **is3D**()	
Argument	None.
Return Value	True: If the programme is 3D.
Description	Confirm whether the programme is 3D or not.
void **set3D**(boolean flag)	
Argument	flag: Set true to indicate the programme is 3D.
Return Value	None.
Description	Manually set the flag which indicating whether the programme is 3D or not.
int **getAudioType**()	
Argument	None.
Return Value	1: Mono 2: Stereo 4: Multi-channel (e.g., 5.1 channel)
Description	Get audio channel information of the programme.
void **setAudioType**(int type)	
Argument	type: 1: Mono 2: Stereo 4: Multi-channel (e.g., 5.1 channel)
Return Value	None.
Description	Manually set the audio channel information.
int **isMultilingual**()	
Argument	None
Return Value	True: If the programme has multi-language capability.
Description	Confirm whether the programme has multi-language capability.

Table III.19 – Method of programme class

void **setMultilungual**(boolean flag)	
Argument	flag: Set true to indicate the programme is multi-language.
Return Value	None.
Description	Manually set the flag which indicating whether the programme is multi-language or not.
java.lang.String[] **getGenre**()	
Argument	None
Return Value	String array which indicating genre of the programme.
Description	Get the string array which indicating the genre of the programme.
void **setGenre**(java.lang.String[] genre)	
Argument	genre: String array which indicating genre of the programme.
Return Value	None.
Description	Set the genre information of the program. It is able to set multiple genres by setting string array.
boolean **hasRecording**()	
Argument	None
Return Value	True: If the recording of the programme is scheduled.
Description	Confirm whether the recording of the programme is scheduled.
ScheduledRecording **getRecording**()	
Argument	None.
Return Value	Recording schedule information.
Description	Get the recording schedule information of the program. When hasRecording() method returns false, this method returns null.
java.lang.String[] **getAudioLanguages**()	
Argument	None.
Return Value	String array indicating audio language.
Description	Get all audio languages included in the programme as string array. Each string is described as the language code of ISO 636.
void **setAudioLanguages**(java.lang.String[] lang)	
Argument	lang: Array of string which indicating the language of audio channel.
Return Value	None.
Description	Manually set all audio languages included in the programme as string array. Each string is described as the language code of ISO 636.
java.lang.String[] **getSubtitleLanguages**()	
Argument	None.
Return Value	String array indicating language of the closed caption.

Table III.19 – Method of programme class

Description	Get all languages information included in the closed caption as string array. Each string is described as the language code of ISO 636.
void **setSubtitleLanguages**(java.lang.String[] lang)	
Argument	lang: Array of string which indicating the language of closed caption.
Return Value	None.
Description	Manually set all closed caption languages included in the programme as string array. Each string is described as the language code of ISO 636.
java.lang.String **getField**(java.lang.String[] fieldId)	
Argument	fieldld: Name of the metadata.
Return Value	String value of specified metadata.
Description	Get the metadata of the programme. This method is applicable to get specific format metadata which is not defined in this Recommendation. (e.g., TV-Anytime, UPnP). In this Recommendation, metadata format and operation are out of scope.
int **getDigitalCopyControlInfo**()	
Argument	None
Return Value	0: Copy free 2: Any limitation of copies (e.g., copy once) 3: Never copy
Description	Get the digital copy control type of the programme.
int **getAnalogCopyControlInfo**()	
Argument	None
Return Value	0: Copy free 3: Never copy
Description	Get the analogue copy control type of the programme.
boolean **isCharged**()	
Argument	None.
Return Value	True: The programme is a pay programme.
Description	Confirm whether the programme is a pay programme or not. In annex [] system, free_CA_mode of EIT will be used.
java.lang.String **getContractVerificationInfo**()	
Argument	None.
Return Value	CA contract information of the programme.
Description	Confirm whether the programme is PPV and/or recording is allowed. In annex [] system, contract_verification_info of EIT or SDT will be used.
AVComponent[] **getComponents**(int componentType)	
Argument	componentType: Component ID (Table III.44).
Return Value	Component information corresponding to specified component ID.

Table III.19 – Method of programme class

Description	Get the information of the component (Video, Audio, Closed caption, superimpose, data) included in the programme.
AVComponent[] **createComponents**(String data)	
Argument	data: Data string of the component acquired by the ContentProvider class.
Return Value	Component information as the result of type conversion from component data string.
Description	Convert the component data sting (Program.COMPONENTS), which is the result of programme search through ContentProvider class, to class instance of the component information.
android.net.Uri **getProgramUri**()	
Argument	None.
Return Value	Intent URI corresponding to the scheduled recording of the programme.
Description	Return the URI to send the intent of scheduled recording when the intent is sent.

III.3.10 ProgramListProvider Class

java.lang.Object

└android.content.ContentProvider

└**java-cable.or.int.media.ProgramListProvider**

This class supports the management of the database of programme information. It provides the functions such as the programme information search, information update, and so on.

III.3.10.1 Labels for database columns

Table III.20 shows the labels for the database column to specify the search query of the ContentProvider class.

Table III.20 – Database labels of programme information

Label name (Programme field)	Type	Values or corresponding class method
ID	Integer	Identifier of each element (rows)
PROGRAM_URI	String	Program#getProgramUri()
NAME	String	Program#getName()
DESCRIPTION	String	Program#getDescription()
LONG_DESCRIPTION	String	Program#getLongDescription()
START_TIME	Integer	Program#getStartTime()
DURATION	Integer	Program#getDuration()
END_TIME	Integer	Same value of Program#getStartTime() + Program#getDuration() (It is defined to help the programme search based on the time)
PROGRAM_ID	Integer	Program#getProgramID()
PROGRAM_ID_TYPE	Integer	Program#getProgramIDType()
CHANNEL	Integer	Program#getChannel() Channel.ID value of the ContentProvider of the channel information shall be set.

Table III.20 – Database labels of programme information

Label name (Programme field)	Type	Values or corresponding class method
EPISODE	Integer	Program#getEpisode()
TOTAL_EPISODES	Integer	Program#getTotalEpisodes()
PARENTAL_RATING	String	The value of Program#getParentalRating() and ParentalRating#getName() shall be set as comma separated format.
LOCKED	Integer	Program#isLocked()
BLOCKED	Integer	Program#isBlocked()
SUBTITLES	Integer	Program#hasSubtitles()
HD	Integer	Program#isHD()
IS_3D	Integer	Program#is3D()
AUDIO_TYPE	Integer	Program#getAudioType()
MULTILINGUAL	Integer	Program#isMultilingual()
GENRE	String	Program#getGenre()
HAS_RECORDING	Integer	Program#hasRecording()
FIELDS	String	Program#getField() Enumerate "[Name]=[Value]" formatted strings with comma separation.
NOTE 1 – Labels are referred as the **Program.[Label name]** format. NOTE 2 – In this table, each parameter originally defined as Boolean is defined as Integer. In this case, value 0 means "true" and value 1 means "false". NOTE 3 – In this table, each parameter originally defined as Array of characters is defined as String. In this case, original characters are connected by a comma and formed one String. NOTE 4 – In this table, the parameter originally defined as java.util.Data is defined as Integer. In this case, "UNIX time" (the elapsed time from 00:00;00 on January 1, 1970) shall be used.		

III.3.10.2 Methods

Methods related to this class are defined in Table III.21.

Table III.21 – ProgramListProvider class method

Program[] **getProgramsByTime**(Date startTime, Date endTime, Channel channel)	
Argument	startTime: Start time of the programme endTime: End time of the programme channel: Channel
Return Value	Array of programme information retrieved by the specified search query.
Description	Retrieve programmes with start/end time. It is possible to specify target channels, but null is allowed if no channel is specified.
Program[] **getCurrentPrograms**(int networkType)	
Argument	networkType: Broadcasting network identifier (Table III.2).
Return Value	Array of the programme information which is currently on the air.

Table III.21 – ProgramListProvider class method

Description	Get the programme information which is currently on the air at the specified broadcasting channel.
Program[] **getProgramsByName**(String query, Channel channel)	
Argument	query: Search query channel: Channel
Return Value	Array of the channel information corresponding to the specified search query.
Description	Retrieve the programmes by using the search query. Targets of the retrieval are programme name, programme description, and programme detailed description. It is possible to specify target channels, but null is allowed if no channel is specified. Also, when only channel is specified and query is set as null, all programme information of the day will be returned.
void **setProgramListEventListener**(ProgramListEventListener listenerClass)	
Argument	listenerClass: Class for the event listener.
Return Value	None.
Description	Set the class specified by the argument as the event listener. This event listener will be called when any event related to the channel information is invoked.

III.3.11 Player class

java.lang.Object

└android.view.View

  └android.view.SurfaceView

    └**java-cable.or.int.media.Player**

This class provides channel change, display control of the broadcasting, and so on. This class does not create class instance from the application side but receives class instance from the method of PlayerControl class.

This class is the subclass of the SurfaceView of Android API, and the usage of control (such as the layout control, size control, etc.) is the same as for it. Therefore, the actual control function of the tuner/decoder is needed to be implemented corresponding to the SurfaceView method.

III.3.11.1 Field

Identifiers and type definitions related to this class are defined in Tables III.22 and III.23.

Table III.22 – Speed of trick play field

Field name	Description
MAX_FORWARD_SPEED	Maximum range of fast forwarding.
MAX_SLOW_SPEED	Maximum range of slow down playback.
MAX_BACKWARD_SPEED	Maximum range of rewind.
MAX_BACKWARD_SLOW_SPEED	Maximum range of slow down rewind.

Table III.23 – URI field

Field name	Description
PLAYER_URI	Intent URI for player control
START_PLAYER_URI	Intent URI to indicate player start.
STOP_PLAYER_URI	Intent URI to indicate player stop.
RELEASE_PLAYER_URI	Intent URI to indicate unallocation of player resource.
CHANNEL_NUMBER_URI	Intent URI to indicate channel change with specific channel number.
PREV_CHANNEL_URI	Intent URI to indicate changing channel to previous one.
NEXT_CHANNEL_URI	Intent URI to indicate changing channel to next one.

The parameter names, which are applicable to putExtra/getExtra for intent, are defined in Table III.24.

Table III.24 – Intent parameter name field

Parameter name	Type	Description
ID	Integer	ID of player which performs channel change.
VOLUME	Integer	Volume of player (0-100).
SUBTITLE	Integer	Display of subtitle 0: None 1: Display subtitle of main language. 2: Display subtitle of sub-language.
SUPERIMPOSE	Integer	Display of superimposed information 0: None 1: Display superimposed information with main language. 2: Display superimposed information with sub-language.
CHANNEL	Integer	Channel number One or two digits (1-12): One button channel change with remote button. Three digits (or four digits if it has sub number): Channel change with three digits.
NETWORK	Integer	Broadcasting network ID. (Table III.2).

III.3.11.2 Method

Methods related to this class are defined in Table III.25.

In this Recommendation, explanations of methods inherited from SurfaceView class are skipped.

Table III.25 – Method of player class

boolean **isFullScreen**()	
Argument	None.
Return Value	True: Full-screen mode.
Description	Confirm whether the player runs full-screen mode.
void **setFullScreen**(boolean flag)	

Table III.25 – Method of player class

Argument	flag: Set true to indicate full-screen output.
Return Value	None.
Description	Indicate the player to output in full-screen mode or not.
int **getPlayState**()	
Argument	None.
Return Value	Player status. 0: Non active 1: Preparing to receive content 2: Receiving and displaying content 3: Halt
Description	Get the status of player.
void **setChannel**(Channel channel)	
Argument	channel: Channel information
Return Value	None.
Description	Change channels specified by the argument.
Channel **getCurrentChannel**()	
Argument	None.
Return Value	Channel information of current channel.
Description	Get current channel information.
Program[] **getPrograms**()	
Argument	None.
Return Value	Array of all channels of specified channel.
Description	Get all programme information of specified channel until end of the day.
void **playContent**(android.net.Uri uri)	
Argument	uri: URI of Channels, local stored content, or remote stored content.
Return Value	None.
Description	Indicate the playback of the content specified by the URI. It is assumed that this method is used for the playback of stored content. However, this method is also applicable to channel change by setting the channel URI as setChannel() method.
void **setChannelByThreeDigit**(int threeDigit)	
Argument	threeDigit: Three digits channel number. (If channel sub-number exists, it is four digits.)
Return Value	None.
Description	Indicate to change current channels to specified one.
void **prevChannel**()	
Argument	None.
Return Value	None.

Table III.25 – Method of player class

Description	Indicate to change channels to previous one.
void **nextChannel**()	
Argument	None.
Return Value	None.
Description	Indicate to change channels to next one.
void **setPlayerEventListener**(PlayerEventListener listenerClass)	
Argument	listenerClass: Class for the event listener.
Return Value	None.
Description	Set the class specified by the argument as the event listener. The event listener will be called when any event related to the player is invoked.
int **getVolume**()	
Argument	None.
Return Value	Volume of the player (0-100).
Description	Get the audio volume of the player.
void **setVolume**(int volume)	
Argument	Volume Volume of the audio player (0-100).
Return Value	None.
Description	Set the audio volume of the player.
void **start**()	
Argument	None.
Return Value	None.
Description	Indicate to start the player.
void **stop**()	
Argument	None.
Return Value	None.
Description	Indicate to stop the player. The assumed usage of this method is to stop content playback temporarily, and so on.
void **release**()	
Argument	None.
Return Value	None.
Description	Indicate the unallocation of the player resource. Any application must call this method when it performs the application suspend, application termination, etc.
void **pause**()	
Argument	None.

Table III.25 – Method of player class

Return Value	None.
Description	Indicate to pause the programme playback/follow-up playback of the programme.
void **play**()	
Argument	None.
Return Value	None.
Description	Indicate to release the pause of the content, and resume the playback/follow-up playback of it. If this method is called during trick play, it indicates to stop trick play and resume the normal speed playback.
int **getPlayPosition**()	
Argument	None.
Return Value	Current playback position of the content.
Description	Get current playback position of the content. This method returns elapsed time (sec) of the content from the beginning. (Count the top of the content as 0 sec.).
int **getDuration**()	
Argument	None.
Return Value	Length of the stored content (sec).
Description	Get the total length of the playing content. In the case of follow-up playback, this method returns the length of the partial content which is already recorded.
void **seek**(int position)	
Argument	position: Seek the playback position (sec).
Return Value	None.
Description	Set the playback position by the agreement (sec). (Count the top of the content as 0 sec.).
void **setSpeed**(int speed)	
Argument	speed: Speed of the fast forwarding/rewind. Plus value indicates fast-forwarding and Minus value indicates rewind. MAX_BACKWARD_SPEED ≦speed ≦ MAX_FORWARD_SPEED
Return Value	None.
Description	Indicate to perform trick-play. If agreement is big, trick play became faster. Value zero indicates the stop of the trick play.
void **setSlowSpeed**(int speed)	
Argument	speed: Speed of the slow playback/Reverse playback. Plus value indicates the slow playback and minus value indicates the slow reverse playback. MAX_SLOW_BACKWARD_SPEED ≦ speed ≦ MAX_SLOW_SPEED
Return Value	None.

Table III.25 – Method of player class

Description	Indicate to perform the slow playback/slow reverse playback. If the argument Is big, content playback becomes slower. Value zero indicates the stop of the trick play.
AVComponent[] **getComponents**(int componentType)	
Argument	componentType: Component identifier (Table III.44).
Return Value	Component information corresponding to the component identifier.
Description	Get each component information (video, audio, subtitle, super impose information, data) included in the current programme.
AVComponent[] **getCurrentActiveComponents**(int componentType)	
Argument	componentType: Component identifier (Table III.44).
Return Value	Component information corresponding to the specified component identifier.
Description	Get the component information from each component (video, audio, subtitle, super impose information, data) included in the current programme if the component is selected.
void **selectComponent**(AVComponent component)	
Argument	component: Component
Return Value	None.
Description	Select the component from the components which included in the programme (video, audio, subtitle, super impose, and data) for play, display, and output.
void **unselectComponent**(AVComponent component)	
Argument	component: Component
Return Value	None.
Description	Select the component from the components which included in the programme (video, audio, subtitle, super impose, and data) as out of scope of play, display, and output.
void **selectComponentByType**(int componentType)	
Argument	componentType: Component Identifier (Table III.44)
Return Value	None.
Description	Indicate the component for play, display, and output by component identifier (C-44).
void **unselectComponentByType**(int componentType)	
Argument	componentType: Component identifier (Table III.44).
Return Value	None.
Description	Indicate the component not for play, display, nor output by component identifier (C-44). This method is applicable to audio mute and so on.
int **getPlayerID**()	
Argument	None.
Return Value	Player ID
Description	Get the player ID.

III.3.12 PlayerControlClass

java.lang.Object

└java-cable.or.int.media.PlayerControl

This class provides management function to the tuner/decoder, and this class provides the creation and management of player class.

III.3.12.1 Method

Methods related to this class are defined in Table III.26.

Table III.26 – PlayerControl class method

Player[] **getPlayers**()	
Argument	None.
Return Value	Class instance of player class.
Description	Get all player class instances of the player where the player can perform video output simultaneously.
int **getNumberOfPlayers**()	
Argument	None.
Return Value	Number of player class instances.
Description	Get the number of player class instances where the player can perfume video output simultaneously.
Player **getDefaultPlayer**()	
Argument	None.
Return Value	Player class instance of the default player.
Description	Get the player class instance of the default player.
void **setDefaultPlayer**(Player player)	
Argument	player: Class instance of player class.
Return Value	None.
Description	Set the specified player as default player.
boolean **getAudioMixingMode**()	
Argument	None.
Return Value	True: All audio outputs of all players will be outputted with mixing. False: Only default player outputs the audio.
Description	Confirm the setting whether the STB performs the audio mixing of all players, or only outputs default player's audio.
void **setAudioMixingMode**(boolean mixing)	
Argument	mixing:
Return Value	None.
Description	Set whether the STB performs audio mixing for multiple players or not.
boolean **is3DDisplayMode**()	

Table III.26 – PlayerControl class method

Argument	None.
Return Value	True: The player can output 3D content.
Description	Confirm whether the player has 3D output capability.
boolean **enable3DDisplayMode**(boolean flag)	
Argument	True: If 3D output is valid when STB receives 3D content.
Return Value	True: If the parameter change is completed. False: If the parameter change is not completed.
Description	Enable or disable the 3D output when the STB receives 3D content. However, if the STB is connected to the TV which does not have 3D capability, this method returns false and the 3D output shall not be valid.

III.3.13 ScheduledRecording class

java.lang.Object

└**java-cable.or.int.media.ScheduledRecording**

This class handles information regarding scheduled recording.

III.3.13.1 Field

Identifiers and type definitions related to this class are defined in Table III.27.

Table III.27 – URI field

Field name	Description
CONTENT_URI	ContentProvider URI of search of the recorded content and scheduled recording (RecordingListProvider).
CONTENT_SCHEDULED_RECORDING_URI	ContentProvider URI which focuses on the search of scheduled recording.
CONTENT_RECORDING_URI	ContentProvider URI which focuses on the search of recorded content.
CONTENT_HOME_RECORDING_URI	ContentProvider URI which focuses on the search of remotely recoded content. In this case, the STB will search all content stored at DLNA DMSs in home network. This intent is only valid when the return value of RecordingListProvider.isCacheEnabled() is true.
RECORD_AT_URI	Intent URI to start the recording from now or specified time.

The parameter names, which are applicable to putExtra/getExtra for intent, are defined in Table III.28

Table III.28 – Parameter name field for Intent

Parameter name	Type	Description
STORAGE_TYPE	Integer	Specify the target storage of recoding. 0: Default local storage 1: Default remote storage (DMS)
START_TIME	Integer	Indicate the recording start time by "UNIX time" (the elapsed time from 00:00:00 on January 1, 1970).
DURATION	Integer	Indicate the recording length (sec).
REPEAT_DAYS	Integer	When weekly periodic recording is set, specify the day of a week by the following ID. (Multiple values are allowed). 0x01: Sunday 0x02: Monday 0x04: Tuesday 0x08: Wednesday 0x10: Thursday 0x20: Friday 0x40: Saturday
THREE_DIGIT	Integer	Three-digit channel number (It becomes four digits if sub number exists). If this parameter is not set, set the channel which the default player selects as for the recording.

III.3.13.2 Constructor

Constructors related to this class are defined in Table III.29.

Table III.29 – Constructor of the ScheduledRecording class

ScheduledRecording()	
Argument	None
Description	Initialize and create the class instance.
ScheduledRecording(Program program)	
Argument	program: Programme information for scheduled recording.
Description	Specify the programme information, and Initialize and create the class instance.
ScheduledRecording(Program program , int repeatDays)	
Argument	program: Program information for scheduled recording repeatDays: When weekly periodic recording is set, specify the day of a week by the following ID. (Multiple values are allowed). 0x01: Sunday 0x02: Monday 0x04: Tuesday 0x08: Wednesday 0x10: Thursday 0x20: Friday 0x40: Saturday
Description	Specify the programme information and repeat recording information (if needed), and initialize and create the class instance.

III.3.13.3 Method

Methods related to this class are defined in Table III.30.

Table III.30 – Method of ScheduledRecording class

int **getStartPadding**()	
Argument	None
Return Value	The length of forward offset time (sec) of the recording start.
Description	Get the length of forward offset time (sec) of the recording start against the specified time such as the programme start time.
void **setStartPadding**(int padding)	
Argument	padding: The length of forward offset time (sec) of the recording start.
Return Value	None
Description	Set the length of forward offset time (sec) of the recording start against the specified time such as the programme start time.
int **getEndPadding**()	
Argument	None
Return Value	The length of backward offset time (sec) of the recording start against the specified stop time.
Description	Get the length of backward offset time (sec) of the recording stop against the specified time such as the programme end time.
void **setEndPadding**(int padding)	
Argument	padding: The length of backward offset time (sec) of the recording start.
Return Value	None
Description	Set the length of backward offset time (sec) of the recording end against the specified time such as the programme end time.
int **getRepeatDays**()	
Argument	None
Return Value	When weekly periodic recording is set, specify the day of a week by the following ID. (Multiple values are allowed). 0x01: Sunday 0x02: Monday 0x04: Tuesday 0x08: Wednesday 0x10: Thursday 0x20: Friday 0x40: Saturday
Description	Get the specified day of the week if the weekly periodic recording is set.
java.lang.String **getName**()	
Argument	None
Return Value	String of the programme name.

Table III.30 – Method of ScheduledRecording class

Description	Get the programme name as the string.
java.lang.String **getDescription**()	
Argument	None
Return Value	String of the programme description.
Description	Get the programme description as the sting.
java.lang.String **getLongDescription**()	
Argument	None
Return Value	String of the detailed programme description.
Description	Get the detailed program description.
java.util.Date **getStartTime**()	
Argument	None
Return Value	Broadcasted time (start time) of recorded programme.
Description	Get the broadcasted time of the recorded programme.
int **getDuration**()	
Argument	None
Return Value	Original length of the recorded programme. (sec)
Description	Get the original length of the recorded programme.
boolean **isSeries**()	
Argument	None
Return Value	True: The programme has series setting.
Description	Confirm whether the programme has series setting.
int **getEpisode**()	
Argument	None
Return Value	Episode number of the programme.
Description	Get the episode number of the recorded programme. In Appendix VII system, episode_number of EIT is set corresponding to [b-ARIB TR-B14].
int **getTotalEpisodes**()	
Argument	None
Return Value	Number of all episodes.
Description	Get the total number of episodes in which the programme is included. In Appendix VII system, last_episode_number of EIT is set corresponding to [b-ARIB TR-B14].
int **getProgramID**()	
Argument	None
Return Value	Program Identifier

Table III.30 – Method of ScheduledRecording class

Description	Get the programme identifier. Programme identifier is the ID to specify the programme information within the DB. In Annex X system, event_id contained in EIT shall be used as defined in [b-ARIB TR-B14].
int **getProgramIDType**()	
Argument	None
Return Value	ProgramTypeId Programme identifier (ID_DVB_EVENT)
Description	Get the programme type identifier (defined in Table III.17) of the programme information.
ParentalRating[] **getParentalRating**()	
Argument	None
Return Value	Parental lock rating information of the programme.
Description	Get parental lock rating information of the programme.
void **addParentalRating**(ParentalRating rating)	
Argument	rating: Parental lock rating information
Return Value	None
Description	Set parental lock rating information to the programme.
DiscInfo **getStorageType**()	
Argument	None
Return Value	Storage information.
Description	Get the storage information to store the content.
void **setStorageType**(DiscInfo storage)	
Argument	storage: Storage information
Return Value	None
Description	Set the storage of programme recording, When recording to the remote storage (RemoteDiscInfo class) is specified, the STB performs the DLNA remote recording and its result will inform through onRemoteRecordingReserved() event listener of RecordingListEventListener interface.

III.3.14 Recording class

java.lang.Object

└java-cable.or.int.media.ScheduledRecording

└java-cable.or.int.media.Recording

III.3.14.1 Field

Identifiers related to this class are defined in Tables III.31and III.32.

Table III.31 – Identifier for recording status

Field name	Value	Description
RECORDING_UNREALIZED	0	Scheduled recording is not registered.
RECORDING_SCHEDULED	1	Recording is scheduled.
RECORDING_DEL_SCHEDULED	2	Scheduled recording is cancelled, (including the case where part or all of the recording is completed).
RECORDING_REC_PRESTART	10	Under the preparation of the recording.
RECORDING_REC_AQUIRING_RESOURCES	11	Under resource allocation to recording (such as storage connection, etc.).
RECORDING_REC_STARTED	12	Recording started.
RECORDING_REC_UPDATED	13	Recording information is updated (such as the change of recording schedule).
RECORDING_REC_COMPLETED	14	Scheduled recording is completed.
RECORDING_REC_PARTIALLY_COMPLETED	15	Scheduled recording is partially completed. (Recording has been manually stopped, storage is disconnected, etc.)
RECORDING_TS_AQUIRING_RESOURCES	21	Under resource allocation to time shift watching (such as storage connection, etc.).
RECORDING_TS_STARTED	22	Time shift watching was started.
RECORDING_ERROR	30	Recording error.
* TS stands for time shift.		

Table III.32 – Recording error identifier

Field name	value	Description
ERROR_REC_RESOURCE_LIMITATION	0	Error caused by the resource limitation (e.g., transcoder).
ERROR_INSUFFICIENT_STORAGE	1	Error caused by the lack of the sufficient space of the storage.
ERROR_TUNER_CONFLICT	2	Error caused by the tuner conflict with other scheduled recordings.
ERROR_REC_DRM_RESTRICTION	10	Error caused by the recording restriction of DRM.
ERROR_REC_UNKNOWN	11	Unknown recording error.
ERROR_TS_RESOURCE_LIMITATION	20	Time shift watching is error caused by the resource limitation.
ERROR_TS_DRM_RESTRICTION	21	Error caused by the time shift watching restriction of DRM.
ERROR_TS_UNKNOWN	22	Unknown time shift watching error.
* TS stands for time shift.		

III.3.14.2 Constructor

This class does not inherit the constructor from "java-cable.or.int.media.ScheduledRecording" class but defines separately.

Constructors related to this class are defined in Table III.33.

Table III.33 – Constructor of recording class

Recording()	
Argument	None
Description	Initialize and create the class instance.
Recording(ScheduledRecording src)	
Argument	src: Recording information of scheduled recording.
Description	Initialize and create class instance based on the scheduled recording information.

III.3.14.3 Method

Methods related to this class are defined in Table III.34.

Table III.34 – Method of recording class

void **setStartPadding**(int padding)	
Argument	padding: Invalid
Return Value	None
Description	This method is inherited from the parent class, but it is not valid in this class.
void **setEndPadding**(int padding)	
Argument	padding: Invalid
Return Value	None
Description	This method is inherited from the parent class, but it is not valid in this class.
void **setStorageType**(DiscInfo storage)	
Argument	storage: Invalid
Return Value	None
Description	This method is inherited from the parent class, but it is not valid in this class.
int **getState**()	
Argument	None
Return Value	Identifier of the recording status (Table III.31).
Description	Get the status of the recording or time shifted content.
int **getError**()	
Argument	None
Return Value	Identifier of the recording error (Table III.32).
Description	Get the error status of recording or time-shifted recording. This method will be used when getState() method returns RECORDING_ERROR.

Table III.34 – Method of recording class

int **getId**()	
Argument	None
Return Value	Identifier of recoded content
Description	Get the recording content identifier. Recorded content identifier is unique ID in the recording information DB. In the local recording, this ID is the same value with the Recording.ID of the RecordingListProvider DB.
boolean **isManual**()	
Argument	None
Return Value	True: Recording is created by manual. False: Recording is created through EPG
Description	Confirm whether the recording schedule is created manually (It means the schedule was created by RecordingListProvider#recordAt() method or Intent of Recording.RECORD_AT_URI). If the schedule is created based on the programme information, this method returns false.
boolean **isDeleteProtected**()	
Argument	None
Return Value	True: The protection against to the automatic deletion is set.
Description	Confirm if the content is protected from automatic deletion of the STB for some reason, such as the lack of storages.
void **setDeleteProtected**(boolean flag)	
Argument	True: Content is protected from automatic deletion.
Return Value	None
Description	Indicate to protect the content from automatic deletion of the STB.
int **getSaveDays**()	
Argument	None
Return Value	Least days of content storing.
Description	Get the least storing days of stored content. However, if no other stored content is allowed to be deleted, the content may be deleted. The application can get the default of this value through SystemConfig#getPVRSaveDays(). The setting of isDeleteProtected() has higher priority than this setting.
void **setSaveDays**(int days)	
Argument	days: Least days of content storing.
Return Value	None
Description	Set the least storing days of stored content. However, if no other stored content is allowed to be deleted, the content may be deleted. The application can set the default of this value through SystemConfig#setPVRSaveDays(). The setting of isDeleteProtected() has higher priority than this setting.
int **getSaveEpisodes**()	
Argument	None

Table III.34 – Method of recording class

Return Value	Minimum number of the episodes to be stored.
Description	Get the setting of the minimum number of episodes to be stored. If the STB stored more series of content that this number, the STB may remove these contents from the oldest one. The application can get the default of this value through SystemConfig#getPVRSaveEpisodes(). The setting of isDeleteProtected() has higher priority than this setting.
void **setSaveEpisodes**(int days)	
Argument	days: Minimum number of the episodes to be stored.
Return Value	None
Description	Set the setting of the minimum number of episodes to be stored. If the STB stored more series of content that this number, the STB may remove these contents from the oldest one. The application can sets the default of this value through SystemConfig#setPVRSaveEpisodes(). The setting of isDeleteProtected() has higher priority than this setting.
boolean **hasSubtitles**()	
Argument	None
Return Value	True: If the programme has subtitles.
Description	Confirm if the programme has subtitles.
java.lang.String[] **getSubtitleLanguages**()	
Argument	None
Return Value	Array of the string to indicate the language of the subtitles.
Description	Get all languages of subtitles included in the programme as a string array. Each string is described as the language code of ISO 636.
boolean **isWideScreen**()	
Argument	None
Return Value	True: If the stored programme is wide screen content.
Description	Confirm whether the programme is wide screen (e.g., 16:9).
boolean **isHD**()	
Argument	None
Return Value	True: If the stored programme is HD content.
Description	Confirm whether the content has higher resolution than 720p.
boolean **is3D**()	
Argument	None
Return Value	True: If the stored program is 3D.
Description	Confirm whether the stored programme is 3D or not.

Table III.34 – Method of recording class

int **getAudioType**()	
Argument	None
Return Value	1: Mono 2: Stereo 4: Multi-channel (e.g., 5.1 channel)
Description	Get audio channel information of the stored content.
int **isMultilingual**()	
Argument	None
Return Value	True: If the stored programme has a multi-language capability.
Description	Confirm whether the stored programme has a multi-language capability.
java.lang.String[] **getAudioLanguages**()	
Argument	None
Return Value	String array indicating audio language.
Description	Get all audio languages included in the stored programme as a string array. Each string is described as the language code of ISO 636.
java.lang.String[] **getGenres**()	
Argument	None
Return Value	String array indicating the genre of the stored programme.
Description	Get the string array indicating the genre of the stored programme.
java.util.Date **getRecordingStartTime**()	
Argument	None
Return Value	Recording star time
Description	Get the actual recording start time with padding.
int **getRecordingDuration**()	
Argument	None
Return Value	Length of the recording (sec)
Description	Get the actual length of the recording with the padding.
android.net.Uri **getContentUri**()	
Argument	None
Return Value	Intent URI corresponding to the playback of the stored content.
Description	Get the URI to send the intent which invokes the playback of the content. When the intent is sent, applications can set the parameters described in Table III.24 as putExtra.
java.lang.String **getField**(java.lang.String[] fieldId)	
Argument	fieldId: Name of the metadata.
Return Value	String value of specified metadata.

Table III.34 – Method of recording class

Description	Get the metadata of the stored programme. This method is applicable to get specific format metadata which is not defined in this Recommendation, (e.g., TV-Anytime, UPnP). In this Recommendation, the metadata format and operation is out of scope.
Bookmark[] **getBookmarks**()	
Argument	None
Return Value	Bookmark information corresponding to the stored programme.
Description	Get the bookmark information, including the history of the resume point, chapter information, etc.
void **setBookmarks**(Bookmark[] bookmarks)	
Argument	bookmarks: Bookmark information corresponding to the stored content.
Return Value	None
Description	Set the bookmark information, including the history of resume point, chapter information, etc.
boolean **isLocked**()	
Argument	None
Return Value	True: If the stored programme is under any parental control.
Description	Return true if the watching of the programme is restricted by any parental control.
boolean **isBlocked**()	
Argument	None
Return Value	True: If the watching of stored programme is restricted by any PIN lock.
Description	Return true if the watching of stored programme is restricted by any PIN lock.

III.3.15 Bookmarkclass

java.lang.Object

└java-cable.or.int.media.Bookmark

This class handles the bookmark information (such as the resume point, chapter information, etc.) to the stored content.

III.3.15.1 Constructor

Constructors related to this class are defined in Table III.35.

Table III.35 – Constructor of bookmark class

Bookmark()	
Argument	None
Description	Initialize and create the class instance.
Bookmark(java.util.Date time, java.lang.String name)	
Argument	time: Times in the program (The top of the content is 0). name: Name of bookmark.
Description	Specify the time information and name of the bookmark, and Initialize and create the class instance.

III.3.15.2 Method

Methods related to this class are defined in Table III.36.

Table III.36 – Method of bookmark class

java.util.Date **getTime**()	
Argument	None
Return Value	time: Times in the program (The top of the content is 0).
Description	Get the time information of the bookmark.
java.lang.String **getName**()	
Argument	None
Return Value	String of the bookmark name.
Description	Get the bookmark name as the string.

III.3.16 RecordingListProviderClass

java.lang.Object

└android.content.ContentProvider

└**java-cable.or.int.media.RecordingListProvider**

This class manages the DB of the stored programme list, including the stored programme and scheduled recording. This class provides the management capability such as the search, updating, and so on. The implementation of the DB is out of the scope of this Recommendation. Therefore, various implementations of data storing, such as the list of automatic gathering of recordings, using cloud storage to create a scheduled recording list, etc., are applicable.

III.3.16.1 Database label

Database labels to specify the queries of the search using the ContentProvider class are shown as above.

Database labels related to this class are defined in Table III.37.

Table III.37 – Database labels of stored content list

Label name	Type	Values or corresponding class method
ID	Integer	Identifier of each element (rows)
STORED_URI	String	Recording#getContentUri()
START_PADDING	Integer	ScheduledRecording#getStartPadding()
END_PADDING	Integer	ScheduledRecording#getEndPadding()
REPEAT_DAYS	Integer	ScheduledRecording#getRepeatDays()
NAME	String	ScheduledRecording#getName()
DESCRIPTION	String	ScheduledRecording#getDescription()
LONG_DESCRIPTION	String	ScheduledRecording#getLongDescription()
START_TIME	Integer	ScheduledRecording#getStartTime()
DURATION	Integer	ScheduledRecording#getDuration()
END_TIME	Integer	Same value of ScheduledRecording#getStartTime() + ScheduledRecording#getDuration() (It is defined to help the program search based on the time)
SERIES	Integer	ScheduledRecording#isSeries()
PROGRAM_ID	Integer	ScheduledRecording#getProgramID()
PROGRAM_ID_TYPE	Integer	ScheduledRecording#getProgramIDType()
CHANNEL	Integer	ScheduledRecording#getChannel() Channel.ID value of the ContentProvider of the channel information shall be set. NOTE – This value is not applied to stored content (Recording class).
EPISODE	Integer	ScheduledRecording#getEpisode()
TOTAL_EPISODES	Integer	ScheduledRecording#getTotalEpisodes()
PARENTAL_RATING	String	The value of ScheduledRecording#getParentalRating() and ParentalRating#getName() shall be set as comma separated format.
RECORDING_ID	Integer	Recording#getId() NOTE – If the scheduled recording (ScheduledRecording class) case, input the program ID of the original content (event_id of EIT).
RECORDING_STATE	Integer	Recording#getState()
RECORDING_ERROR	Integer	Recording#getError()
MANUAL	Integer	Recording#isManual()
DELETE_PROTECTED	Integer	Recording#isDeleteProtected()
SAVE_DAYS	Integer	Recording#getSaveDays()
SAVE_EPISODES	Integer	Recording#getSaveEpisodes()
LOCKED	Integer	Recording#isLocked()
BLOCKED	Integer	Recording#isBlocked()
SUBTITLES	Integer	Recording#hasSubtitles()
HD	Integer	Recording#isHD()
IS_3D	Integer	Recording#is3D()

Table III.37 – Database labels of stored content list

Label name	Type	Values or corresponding class method
WIDESCREEN	Integer	Recording#isWideScreen()
AUDIO_TYPE	Integer	Recording#getAudioType()
MULTILINGUAL	Integer	Recording#isMultilingual()
GENRES	String	Recording#getGenres()
FIELDS	String	Recording#getField() Enumerate "[Name]=[Value]" formatted strings with the comma separation.
BOOKMARKS	String	Recording#getBookmarks() Enumerate "[Bookmark name]=[time (from the top of the content)]" formatted strings with comma separation.
NOTE 1 – Labels are referred as the **Recording.[Label name]** format. NOTE 2 – In this table, each parameter originally defined as Boolean is defined as Integer. In this case, value 0 means "true" and value 1 means "false". NOTE 3 – In this table, each parameter originally defined as Array of characters is defined as String. In this case, original characters are connected by comma and formed one String. NOTE 4 – In this table, parameter originally defined as java.util.Data is defined as Integer. In this case, "UNIX time" (the elapsed time from 00:00;00 on January 1, 1970) shall be used.		

III.3.16.2 Method

Methods related to this class are defined in Table III.38.

Table III.38 – Method of RecordingListProvider class

boolean **isCacheEnabled**()	
Argument	None
Return Value	True: If the automatic gathering of the remote recording list.
Description	Return true if the STB automatically gathers the stored content information of DMSs in home networks and stores this information to the local DB. If this function is active, applications can search both of the local stored content and remote stored content through the searchRecordingsByName() method at once.
boolean **getRemoteSearchOrder**(java.lang.String fieldId)	
Argument	fieldId: Field name of searched objects.
Return Value	True: Search order is ascending order. False: Search order is descending order.
Description	Confirm whether the search order is ascending order or descending order to the field name (e.g., "data") of search target of the remote recording.
void **setRemoteSearchOrder**(java.lang.String fieldId, boolean ascending)	
Argument	fieldId: Field name of searched object ascending: True: Search order is ascending order. False: Search order is descending order.
Return Value	None

Table III.38 – Method of RecordingListProvider class

Description	Indicate ascending order or descending order as the search order to the field name (e.g., "data") of search target of the remote recording. If the returning value of isCacheEnabled() is false, only the remote storage which was indicated by the search query is applied to the search result. On the other hand, if isCacheEnabled() returns true, STB can search internal cache.
ScheduledRecording[] **getScheduledRecordings**()	
Argument	None
Return Value	Array of scheduled recording information (including remote recording).
Description	Get the all information of scheduled recording.
void **searchRecordingsByName**(String query, DiscInfo storage)	
Argument	query: Search query storage: Storage information to search
Return Value	None
Description	Perform the search of all stored program whose name, description or detailed description are matched with search query. If null is set as the storage information, all local recording will be searched. If the return value of isCacheEnabled() is true, all remote storage in the home network will be searched. The search result will be informed through the onSearchRecording() event listener (RecordingListEventListner interface).
DiscInfo[] **getLocalStorage**()	
Argument	None
Return Value	Array of the local storage.
Description	Get information of all local storage connected to the STB.
void **searchRemoteStorage**()	
Argument	None
Return Value	None
Description	Send search request of all remote storage (DLNA DMS) on the home network. The search result will be informed through the on SearchStorage() event listener (RecordingListEventListener interface).
DiscInfo **getDefaultStorage**()	
Argument	None
Return Value	Default local storage information.
Description	Get the default local storage information connected to the STB.
RemoteDiscInfo **getDefaultRemoteStorage**()	
Argument	None
Return Value	Default remote storage information.
Description	Get the default remote storage information connected to the STB.

Table III.38 – Method of RecordingListProvider class

void **setDefaultStorage**(DiscInfo storage)	
Argument	storage: Local storage information set as default.
Return Value	None
Description	
void **setDefaultRemoteStorage**(RemoteDiscInfo storage)	
Argument	storage: Remote storage information
Return Value	None
Description	Set the specified remote storage to default remote storage of the STB.
ScheduledRecording **record**(Program program, int storageType)	
Argument	program: Program information for the scheduled recording. storageType: Storage information for the scheduled recording.
Return Value	Information of newly registered scheduled recording.
Description	Schedule the recording of the specified program to the specified storage. If the storage is remote storage (RemoteDiscInfo class), perform the DLNA remote recording and inform its result through the onRemoteRecordingReserved() event listener (RecordingListEventListener interface).
ScheduledRecording **recordAt**(java.util.Date startTime, int duration, int repeatDays, int storageType)	
Argument	startTime: Time to start the recording (If the past time is set, start recording immediately.). duration: Recording length. repeatDays: When weekly periodic recording is set, specify the day of a week by the following ID. (Multiple values are allowed). 0x01: Sunday 0x02: Monday 0x04: Tuesday 0x08: Wednesday 0x10: Thursday 0x20: Friday 0x40: Saturday
Return Value	Information of newly registered scheduled recording.
Description	Schedule the recording by the specified time, Including periodic recording, to the specified storage. If the storage is remote storage (RemoteDiscInfo class), perform the DLNA remote recording and inform its result through the onRemoteRecordingReserved() event listener (RecordingListEventListener interface).
void **removeScheduledRecording**(ScheduledRecording recording)	
Argument	recording: Scheduled recording information to be cancelled.
Return Value	None
Description	Remove the specified scheduled recording information and cancel the schedule.

Table III.38 – Method of RecordingListProvider class

void **stopRecording**(Recording recording)	
Argument	recording: Scheduled recording information to be stopped.
Return Value	None
Description	Stop the specified scheduled recording when the recording is running.
void **refreshRecordingList**()	
Argument	None
Return Value	None
Description	Refresh the recording list. Local recording information will be updated to the latest one and the cache of remote recording information will be cleared. (The latest information will be acquired by new search.)
void **setRecordingListEventListener**(RecordingListEventListener listenerClass)	
Argument	listenerClass: Class for the event listener.
Return Value	None
Description	Set the class specified by the argument as the event listener. This event listener will be called when any event related to the recording information.

III.3.17 ChannelListEventListener interface

java-cable.or.int.media.event.ChannelListEventListener

In this section, the methods of the event listener receiving the event regarding channel information are defined.

III.3.17.1 Method

Methods related to this interface are defined in Table III.39.

Table III.39 – Method of ChannelListEventListener interface

void **onChannelScan**()	
Argument	None
Description	Inform the completion of channel scan.
void **onChannelListUpdate**()	
Argument	None
Description	Inform any changes of the list of channel information are arisen.

III.3.18 ProgramListEventListener interface

java-cable.or.int.media.event.ProgramListEventListener

In this clause, the methods of the event listener receiving the event regarding programme information are defined.

III.3.18.1 Method

Methods related to this interface are defined in Table III.40.

Table III.40 – Method of ProgramListEventListener interface

void **onUpdateProgramList**()	
Argument	None
Description	Inform any changes of the list of channel information are arisen.

III.3.19 PlayerEventListener interface

java-cable.or.int.media.event.PlayerEventListener

In this clause, the methods of the event listener receiving the event regarding the player (tuner, decoder) are defined.

III.3.19.1 Method

Methods related to this class are defined in Table III.41.

Table III.41 – Method of PlayerEventListenerInterface

void **onPlayStateChange**(int state, int error)	
Parameters	state: State of Player 0: Player stop 1: Player start (State before display start) 2: Player active error: Error in Player (refer to onChannelChangeError())
Description	Notify state change of Player
void **onChannelChangeError**(Channel channel, int errorState, int errorCode)	
Parameters	channel: Channel information for channel selection request (or error caused) errorState: State of error 0: Channel does not relate to tuner 1: Not available for stream reception (weak signal level, etc.) 2: Using tuner (or occupied by other application) 3: Assigned channel is locked by parental control. 4: Decoding is not available in encrypted channel. (No insertion of B-CAS/C-CAS cards, etc.) 5: Unknown channel 6: Interruption of channel change (Break in by other process, etc.) 7: Impossible channel change due to recording process 8: URI is not reachable. 9: Impossible playback due to small bandwidth (mainly for watching remote recorded programme) 10: Impossible channel change by forward/back channel key (incorrect channel mask setting by favourite list) 11: Impossible playback due to resource shortage (Codec error, etc.) 12: No existence of desired channel in transport stream 100: Error is not defined. errorCode: Error code depended on broadcast system (Double-octed error codes described in 4.14, Chapter 2 of [b-ARIB TR-B15], in case of Japan)
Description	Notification of caused error during channel change
void **onChannelChangeSucceeded**(Channel channel)	
Parameters	channel: Channel information done by channel selection request
Description	Notifies successful completion of channel change

Table III.41 – Method of PlayerEventListenerInterface

void **onFullScreenChange**()	
Parameters	None
Description	Notifies change to full-screen
void **onProgramsChanged**()	
Parameters	None
Description	Notifies that change and/or renewal of programme information is occurred in watching channel.
void **onParentalRatingChanged**(Program program, ParentalRating rating, java.lang.String DRMSystemID, boolean blocked)	
Parameters	program: Program information rating: Status of parental audio visual control DRMSystemID: Status of DRM (CAS) blocked: True if it is audio visual block object
Description	Notifies that condition of parental control changes due to change of program in watching channel.
void **onParentalRatingError**(Program program, ParentalRating rating, java.lang.String DRMSystemID)	
Parameters	program: Program information rating: Status of parental audio visual control DRMSystemID: DRM (CAS) Parameters urn:arib:casystemid:3:C-CAS [b-JCL SPEC-005] urn:arib:casystemid:4: C-CAS [b-JCL SPEC-005] urn:arib:casystemid:5: B-CAS [b-ARIB TR-B14/B15] urn:arib:casystemid:6: C-CAS [b-JCL SPEC-005] urn:arib:casystemid:7: C-CAS [b-JCL SPEC-001-01, 002, 007]
Description	Notifies error if restricted programme by parental audio visual control is to be displayed or replayed.
void **onDRMRightsError**(int errorState, int errorCode, Program program, java.lang.String DRMSystemID, java.lang.String rightsIssuerURL)	
Parameters	errorState: Status of error 0: Replay is not available because of no DRM (CAS) 1: Replay is not available because of no DRM (CAS) 2: Replay is available because of correct DRM (CAS) errorCode: Error code broadcast method (Hexadecimal number is applied as error code described in clause 5.15, Section 5 of [b-ARIB TR-B14]) program: Program information DRMSystemID: Refer to onParentalRatingError() rightsIssuerURL: URL of DRM (CAS) error related information (void if it is none)
Description	Notifies DRM (CAS) status change
void **onBmlError**(int errorCode)	
Parameters	errorCode: Status of error 400: BML document is not found. 401: No matching of versions between BML document and Engine 402: Implementation error of BML display and tracking error of external data
Description	Notifies error caused in BML process

III.3.20 RecordingListEventListener Interface

java-cable.or.int.media.event.RecordingListEventListener

This clause defines methods of the event listener which receives an event when recorded programme-related information event has occurred.

III.3.20.1 Method

Methods related to this interface are defined in Table III.42.

Table III.42 – Method of RecordingListEventListener interface

void **onSearchRecording**(ScheduledRecording[] recordings)	
Parameters	recordings: Search result of recorded program
Description	Notifies search result of recorded program
void **onSearchStorage**(DiscInfo[] storages)	
Parameters	storages: Search result of storage
Description	Notifies search result of storage (mainly for remote storage (DLNA DMS))
void **onPVREvent**(int recordingState, Recording recording)	
Parameters	recordingState: Justifies value of recorded status (Table III.31) recording: Recording information
Description	Notifies change status for recording information
void **onRemoteRecordingReserved**(boolean isSuccess, ScheduledRecording recording)	
Parameters	isSuccess: True if reserved recording is successful recording: Recording information
Description	Notifies that reservation of remote recording is successful or not.

III.3.21 Broadcast Intent

Permissible events, such as broadcast intent, are as below. There is no parameter to be transferred by getExtra in any case.

Broadcast intent names used in this interface are defined in Table III.43.

Table III.43 – List of broadcast intent

Intent Name	Content
java-cable.or.int.intent.action.CHANNEL_SCAN	Notifies completion of channel scan Equivalent to ChannelListEventListener#onChannelScan()
java-cable.or.int.intent.action.CHANNEL_LIST_UPDATE	Notifies channel list change Equivalent to ChannelListEventListener#onChannelListUpdate()
java-cable.or.int.intent.action.UPDATE_PROGRAM_LIST	Notifies programme list change Equivalent to ProgramListEventListener#onUpdateProgramList()

III.3.1 AVComponent Interface

Defines common template of each class that deals with component information for video, audio, subtitle, super-impose, and data broadcast (BML).

III.3.21.1 Fields

Fields related to this interface are defined in Table III.44.

Table III.44 – Component descriptor

Field name	Value	Description
COMPONENT_TYPE_VIDEO	0	Video component
COMPONENT_TYPE_AUDIO	1	Audio component
COMPONENT_TYPE_SUBTITLE	2	Subtitle component
COMPONENT_TYPE_SUPERIMPOSE	3	Superimpose component
COMPONENT_TYPE_DATA_CONTENT	4	Data broadcast (BML) component
COMPONENT_TYPE_ALL	10	All components

III.3.21.2 Method

Methods related to this interface are defined in Table III.45.

Table III.45 – Method of AVComponent interface

int **getComponentTag**()	
Parameters	None
Returns	Component tag value
Description	Gets component tag value in MPEG-2 TS to identify type of component
int **getPid**()	
Parameters	None
Returns	PID value
Description	Gets PID value in MPEG-2 TS that relates component
int **getType**()	
Parameters	None
Returns	Component description value (Table III.44)
Description	Gets component description value
java.lang.String **getEncoding**()	
Parameters	None
Returns	Descriptor of encode method
Description	Gets encode method as strings satisfying OIPF MEDIA2 specification
boolean **isEncrypted**()	
Parameters	None
Returns	True if it is encrypted component
Description	Gets information that components are encrypted or not.
int **getComponentType**()	
Parameters	None
Returns	Component type value of EIT
Description	Gets component type value directly included in EIT component descriptor specified in [b-ARIB TR-B14]

III.3.22 AVAudioComponent Class

java.lang.Object

└**java-cable.or.int.media.param.AVAudioComponent** (implements AVComponent)

Methods related to this class are defined in Table III.46.

Table III.46 – Method of AVAudioComponent class

java.lang.String **getLanguage**()	
Parameters	None
Returns	Character string showing audio component language
Description	Gets character string as an audio component language. String is described by ISO 639 language code
java.lang.String **getLanguage2**()	
Parameters	None
Returns	Character string showing sub-audio component language
Description	Gets character string as a sub- audio component language. String is described by ISO 639 language code
boolean **hasAudioDescription**()	
Parameters	None
Returns	True if it is included audio for visually handicapped person
Description	Confirms whether audio for visually handicapped person is contained in audio component
int **getAudioChannels**()	
Parameters	None
Returns	Number of audio channel (ex. 2 for stereo, 5 for 5.1 ch surround)
Description	Gets audio channel number contained in audio component
int **getStreamType**()	
Parameters	None
Returns	Component type value of EIT
Description	Gets stream type value directly in PMT described in [b-ARIB TR-B14]

This portion deals with the audio component. The difference with the AVComponent interface is described.

III.3.23 AVVideoComponent class

java.lang.Object

└**java-cable.or.int.media.param.AVVideoComponent** (implements AVComponent)

This portion deals with the video component. The difference with the AVComponent interface is described.

Methods related to this interface are defined in Table III.47.

Table III.47 – Method of AVVideoComponent class

double **getAspectRatio**()	
Parameters	None
Returns	Aspect ratio (ex.1.33 for 4:3, 1.78 for 16:9)
Description	Gets video component aspect ratio as decimal fraction that horizontal length divided by longitudinal length

III.3.24 AVSubtitleComponent class

java.lang.Object

└**java-cable.or.int.media.param.AVSubtitleComponent** (implements AVComponent)

This portion deals with the title component. The difference with the AVComponent interface is described.

Methods related to this interface are defined in Table III.48.

Table III.48 – Method of AVSubtitleComponent class

java.lang.String **getLanguage**()	
Parameters	None
Returns	Character string showing title component language
Description	Gets character string as a title component language. String is described by ISO 639 language code
java.lang.String **getLanguage2**()	
Parameters	None
Returns	Character string showing sub-title component language
Description	Gets character string as a sub-title component language. String is described by ISO 639 language code
boolean **isHearingImpaired**()	
Parameters	None
Returns	True if it is included title for auditory handicapped person
Description	Confirms whether title for auditory handicapped person is contained in title component

III.3.25 AVDataContentComponent class

java.lang.Object

└**java-cable.or.int.media.param.AVDataContentComponent** (implements AVComponent)

This portion deals with the data broadcast (BML) component. The difference with the AVComponent interface is described.

Methods related to this interface are defined in Table III.49.

Table III.49 – Method of AVDataContentComponent class

boolean **isContentAssociated**()	
Parameters	None
Returns	True if it is related programme content directly to data broadcast content
Description	Confirms that the content of the data broadcast component is directly related to programme content. If it returns false, some generic information such as broadcast centre information, weather news and traffic information etc., are usually described instead of programme information.
int **getDigitalCopyControlInfo**()	
Parameters	None
Returns	0 for copy free 2 for copyonce 3 for copynever
Description	Gets digital copy restriction value on data broadcast component.

III.3.26 DiscInfo Class

java.lang.Object

└java-cable.or.int.media.param.DiscInfo

This portion deals with information on storage. In relation to recording information, viewing reservation information is treated in this class.

III.3.26.1 Field

Identifiers related to this interface are defined in Table III.50.

Table III.50 – Descriptor for storage class

Field Name	Value	Contents
LOCAL_STORAGE	0	Represent local storage (embedded storage, USB connected storage, etc.)
VIEWING_RESERVATION	1	Represent viewing reservation
DLNA_UPLOAD	2	Represent DLNA upload recording or storage over DMS

III.3.26.2 Method

Methods related to this interface are defined in Table III.51.

Table III.51 – Method for DiscInfo class

int **getStorageType**()	
Parameters	None
Returns	Descriptor for Storage class (Table III.50)
Description	Confirms storage class
double **getTotalStorageSize**()	
Parameters	None
Returns	Total storage capacity (in Byte)
Description	Gets total storage capacity

Table III.51 – Method for DiscInfo class

double **getFreeStorageSize**()	
Parameters	None
Returns	Vacant storage capacity (in Byte)
Description	Gets vacant storage capacity
double **getReservedStorageSize**()	
Parameters	None
Returns	Storage capacity (in Byte) saved by reserved storage
Description	Gets storage capacity saved by reserved storage
java.lang.String **getName**()	
Parameters	None
Returns	Storage name
Description	Gets storage name set in storage
boolean **setName**(java.lang.String name)	
Parameters	name: Storage name
Returns	True, if it is modified, false, if it cannot be modified
Description	If it can be modified, change storage name set in storage

III.3.27 RemoteDiscInfoClass

java.lang.Object

└java-cable.or.int.media.param.DiscInfo

└**java-cable.or.int.media.param.RemoteDiscInfo**

In the information related to storage, remote recording information is described. These are used for both DLNA upload recording, and DMS content search and replay. Difference with DiscInfo only is described below.

III.3.27.1 Method

Methods related to this interface are defined in Table III.52.

Table III.52 – Method of RemoteDiscInfoClass

int **getDeviceHandle**()	
Parameters	None
Returns	Device handle(UPnP Object ID)
Description	Gets DMS device handle value
int **getUDN**()	
Parameters	None
Returns	UDN value (uuid)
Description	Gets DMS uuid value
android.net.Uri **getProviderUri**()	
Parameters	None
Returns	ContentProvider URI for programme search in DMS
Description	Gets ContentProvider URI for searching recorded programme storaged in DMS

Table III.52 – Method of RemoteDiscInfoClass

java.lang.String **getIpAddress**()	
Parameters	None
Returns	Character set denotes IP address
Description	Gets DMS IP address
java.lang.String **getMACAddress**()	
Parameters	None
Returns	Character set denotes MAC address
Description	Gets DMS MAC address
java.lang.String **getSortCAP**()	
Parameters	None
Returns	Character set denotes DMS sorting capacity
Description	Confirms DMS sorting capacity
java.lang.String **getSearchCAP**()	
Parameters	None
Returns	Character set denotes search query related to DMS
Description	Confirms search query related to DMS
java.lang.String **getDlnaCAP**()	
Parameters	None
Returns	Character set denotes DMS search capacity
Description	Confirms other search capacity of DMS defined in DLNA 1.5
java.lang.String **getJava-cableCAP**()	
Parameters	None
Returns	Character set (A-F descriptors) denotes DMS search capacity
Description	Gets DMS search capacity

III.4 Detailed setting descriptions for API package java-cable.or.int.config

III.4.1 General

API package java-cable.or.int.config is an API assembly enabling configuration of the STB system and processing message from Java applications. In specific, the following functions are included.

- System configuration (SystemConfig class): API assembly providing the configuration function of the whole system
- System message (SystemMessage class): API assembly providing read-write functions for the system message

Figure III.10 depicts the API class structure defined in the package.

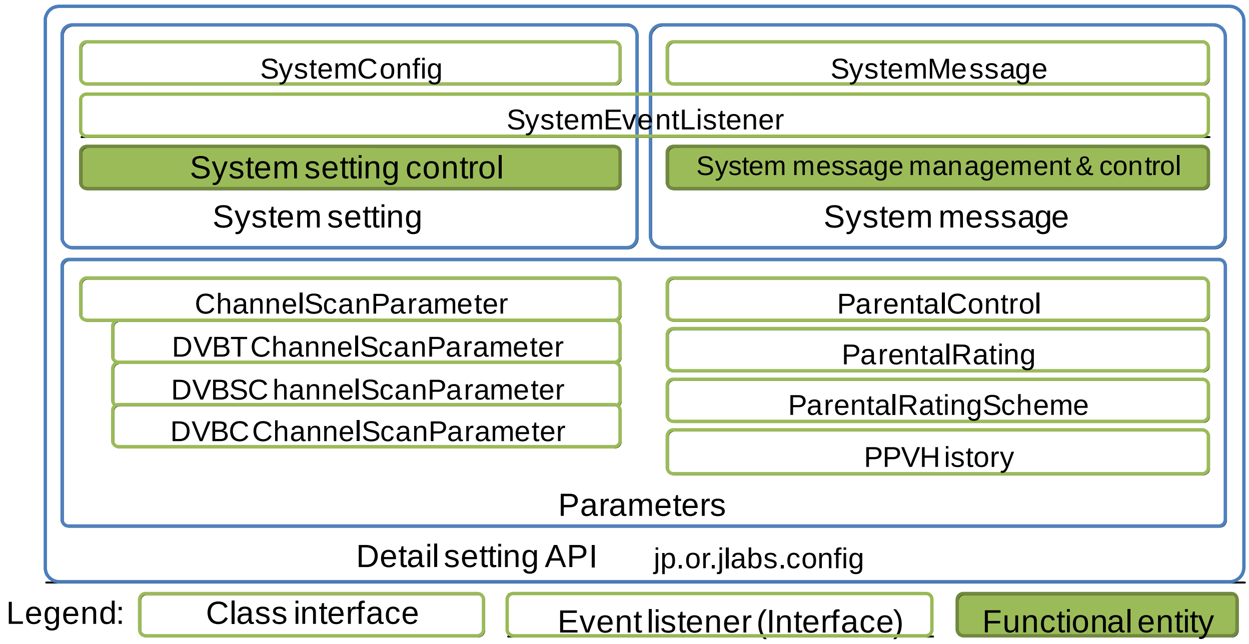


Figure III.10 – API class structure for detailed configuration

III.4.2 SystemConfig Class

java.lang.Object

└**java-cable.or.int.config.SystemConfig**

Class providing the detailed configuration function of whole system

III.4.2.1 Field

Identifiers and type definitions related to this interface are defined in Tables III.53, III.54 and III.55.

Table III.53 – System information field

Field Name	Type	Meaning
DEVICE_ID	String	Device ID of STB (STB ID)
DEVICE_VENDOR_NAME	String	Manufacture (Vender) name of STB
DEVICE_MODEL_NAME	String	Model name of STB
DEVICE_SERIAL_NUMBER	String	Serial number of STB
DEVICE_SOFTWARE_VERSION	String	Software version of STB
DEVICE_HARDWARE_VERSION	String	Hardware version of STB
JAVA-CABLE_API_VERSION	Integer	Version of API specification (Satisfying this specification: 1)

Table III.54 – Descriptors for remote controller key event

Field Name	Value	Meaning
RCTYPE_CALLBACK	1	Gets remote controller key event by call back
RCTYPE_KEYBOARD	2	Gets remote controller key event as Android key event

Table III.55 – Mapping with remote controller key and keyboard

Name of Remote Controller Key Label	Meaning
RCKEYCODE_POWER	Power ON/OFF of STB
RCKEYCODE_1	Channel Key 1
RCKEYCODE_2	Channel Key 2
RCKEYCODE_3	Channel Key 3
RCKEYCODE_4	Channel Key 4
RCKEYCODE_5	Channel Key 5
RCKEYCODE_6	Channel Key 6
RCKEYCODE_7	Channel Key 7
RCKEYCODE_8	Channel Key 8
RCKEYCODE_9	Channel Key 9
RCKEYCODE_10	Channel Key10
RCKEYCODE_11	Channel Key 11
RCKEYCODE_12	Channel Key 12
RCKEYCODE_CHANNEL_UP	Before Channel Key
RCKEYCODE_CHANNEL_DOWN	After Channel Key
RCKEYCODE_LEFT	Cross Key Left
RCKEYCODE_UP	Cross Key Up
RCKEYCODE_RIGHT	Cross Key Right
RCKEYCODE_DOWN	Cross Key Down
RCKEYCODE_OK	OK Key (Enter Key)
RCKEYCODE_BACK	Return Key (Back Key)
RCKEYCODE_MENU	Menu Key
RCKEYCODE_SEARCH	Search Key
RCKEYCODE_VOL_UP	Sound Volume Key (Loud)
RCKEYCODE_VOL_DOWN	Sound Volume Key (Low)
RCKEYCODE_VOL_MUTE	Mute Key
RCKEYCODE_DATA_CONTENT	Data Broadcast Key
RCKEYCODE_MULTILINGUAL	Language Change (Sub-audio) Key
RCKEYCODE_SUBTITLE	Sub-title Key
RCKEYCODE_NETWORK_DTT	Terrestrial Digital Broadcast Key
RCKEYCODE_NETWORK_BS	BS Digital Broadcast Key
RCKEYCODE_NETWORK_CATV	CS Digital Broadcast Key
RCKEYCODE_HOME	Home Key
RCKEYCODE_PORTAL	Portal Key
RCKEYCODE_VOD	VOD Key
RCKEYCODE_CHANGE_DISPLAY_MODE	Input Change Key
RCKEYCODE_PLAY	Play Key
RCKEYCODE_PAUSE	Pause Key
RCKEYCODE_STOP	Stop Key

Table III.55 – Mapping with remote controller key and keyboard

Name of Remote Controller Key Label	Meaning
RCKEYCODE_REC	Record Key
RCKEYCODE_FORWARD	Forward Key
RCKEYCODE_REWIND	Rewind Key
RCKEYCODE_PREV	Previous Key
RCKEYCODE_NEXT	Next Key
RCKEYCODE_ARIB_BLUE	Data Broadcast Control Key Blue
RCKEYCODE_ARIB_RED	Data Broadcast Control Key Red
RCKEYCODE_ARIB_GREEN	Data Broadcast Control Key Green
RCKEYCODE_ARIB_YELLOW	Data Broadcast Control Key Yellow
RCKEYCODE_EXT_0	Maker Specific Extension Key 0
RCKEYCODE_EXT_1	Maker Specific Extension Key 1
RCKEYCODE_EXT_2	Maker Specific Extension Key 2
RCKEYCODE_EXT_3	Maker Specific Extension Key 3
RCKEYCODE_EXT_4	Maker Specific Extension Key 4
RCKEYCODE_EXT_5	Maker Specific Extension Key 5
RCKEYCODE_EXT_6	Maker Specific Extension Key 6
RCKEYCODE_EXT_7	Maker Specific Extension Key 7
RCKEYCODE_EXT_8	Maker Specific Extension Key 8
RCKEYCODE_EXT_9	Maker Specific Extension Key 9

III.4.2.2 Method

Methods related to this interface are defined in Table III.56.

Table III.56 – Method of SystemConfig Class

void **factoryReset**()	
Parameters	None
Returns	None
Description	Returns default at factory shipping
boolean **isPVREnabled**()	
Parameters	None
Returns	True, if recording function is effective
Description	Confirms whether STB recording function is effective or not. For example, in case of no embedded storage STB, it becomes false if it is not available for external USB storage or upload recording to DLNA DMS.
boolean **isSystemReady**()	
Parameters	None
Returns	True if broadcast function (Tuner, etc.) is effective.
Description	Confirms whether broadcast function (Tuner, etc.) is effective or not.

Table III.56 – Method of SystemConfig Class

java.lang.String **getPostCode**()	
Parameters	None
Returns	Setting value for postal code
Description	Gets postal code set in STB
void **setPostCode**(java.lang.String code)	
Parameters	code: Character set denotes postal code
Returns	None
Description	Sets postal code into STB
java.lang.String **getLocalArea**()	
Parameters	None
Returns	Name of local area
Description	Gets name of local area set in STB
void **setLocalArea**(java.lang.String area)	
Parameters	area: Character set denotes name of local area
Returns	None
Description	Sets name of local area into STB
boolean **isAutoSoftwareUpdateEnabled**()	
Parameters	None
Returns	True if automatic software update is effective
Description	Confirms whether automatic software update is effective or not
void **enableAutoSoftwareUpdate**(boolean flag)	
Parameters	flag: True in case of effecting automatic software update
Returns	None
Description	Sets whether automatic software update to be effective or not
boolean **isChannelMaskEnabled**()	
Parameters	None
Returns	True if channel mask is effective
Description	Confirms whether channel mask function of STB is effective or not
void **enableChannelMask**(boolean flag)	
Parameters	flag: True in case of effecting channel mask
Returns	None
Description	Sets whether channel mask function of STB to be effective or not
java.lang.String **getPreferredAudioLanguage**()	
Parameters	None
Returns	Priority language of audio
Description	Gets ISO 639 language code to be selected as priority language in audio replay. Several languages are described from high priority order segregated with comma.

Table III.56 – Method of SystemConfig Class

void **setPreferredAudioLanguage**(java.lang.String lang)	
Parameters	lang: Priority language of audio
Returns	None
Description	Sets ISO 639 language code to be selected as priority language in audio replay. As parameters several languages are described from high priority order segregated with comma.
java.lang.String **getPreferredSubtitleLanguage**()	
Parameters	None
Returns	Priority language of subtitle
Description	Gets ISO 639 language code (ex. "jpn" for Japan) to be selected as priority language in subtitle display. Several languages are described from high priority order segregated with comma.
void **setPreferredSubtitleLanguage**(java.lang.String lang)	
Parameters	lang: Priority language of subtitle
Returns	None
Description	Sets ISO 639 language code to be selected as priority language in subtitle display. As parameters several languages are described from high priority order segregated with comma.
java.lang.String **getCountryId**()	
Parameters	None
Returns	Country code
Description	Gets country code in area where STB is to be used using ISO 3166 country code
void **setCountryId**(java.lang.String country)	
Parameters	country: Country code
Returns	None
Description	Sets country code in area where STB is to be used using ISO 3166 country code
int **getDefaultAudioMode**()	
Parameters	None
Returns	0: Simultaneous playback of main and sub audio 1: Playback of main audio 2: Playback of sub audio
Description	Gets STB default audio playback mode
void **setDefaultAudioMode**(int mode)	
Parameters	mode: Audio mode 0: Simultaneous playback of main and sub audio 1: Playback of main audio 2: Playback of sub audio
Returns	None
Description	Sets STB default audio playback mode

Table III.56 – Method of SystemConfig Class

int **getDefaultSubtitleMode**()	
Parameters	None
Returns	0:No subtitle display 1: Subtitle display of main language 2: Subtitle display of sub language
Description	Gets STB default subtitle display mode
void **setDefaultSubtitleMode**(int mode)	
Parameters	mode: Subtitle display mode 0: No subtitle display 1: Subtitle display of main language 2: Subtitle display of sub language
Returns	None
Description	Sets STB default subtitle display mode
int **getDefaultSuperimposeMode**()	
Parameters	None
Returns	0: No superimpose display 1: Superimpose display of main language 2: Superimpose display of sub language
Description	Gets STB default superimpose display mode
void **setDefaultSuperimposeMode**(int mode)	
Parameters	mode: Superimpose display mode 0: No superimpose display 1: Superimpose display of main language 2: Superimpose display of sub language
Returns	None
Description	Sets STB default superimpose display mode
double **getTvAspectRatio**()	
Parameters	None
Returns	Aspect ratio (ex. 1.33 for 4:3, 1.78 for 16:9)
Description	Gets set value of aspect ratio of output screen in composite output as decimal of length divided by width
void **setTvAspectRatio**(double ratio)	
Parameters	ratio: Aspect ratio (ex. 1.33 for 4:3, 1.78 for 16:9)
Returns	None
Description	Sets aspect ratio of output screen in composite output as decimal of length divided by width
java.lang.String **getSpdifMode**()	
Parameters	None
Returns	"AUTO": Automatic mode selection "PCM": PCM output fixed mode
Description	Gets set value of audio output mode of S/PDIF terminal

Table III.56 – Method of SystemConfig Class

void **setSpdifMode**(java.lang.String mode)	
Parameters	mode: Audio output mode "AUTO": Automatic mode selection "PCM": PCM output fixed mode
Returns	None
Description	Sets audio output mode of S/PDIF terminal
int **getPVRPolicy**()	
Parameters	None
Returns	Recording storage policy value 0: Holds recorded programme (Additional recording is not permitted until capacity saved by manual deletion) 1: Deletes preferentially from past watched programme 2: Deletes preferentially from old programme (follow setting of getPVRSaveDays(), getPVRSaveEpisodes())
Description	Gets preferential policy for storage and deletion of recorded programme
void **setPVRPolicy**(int policy)	
Parameters	policy: value of recording storage policy 0: Holds recorded programme (Additional recording is not permitted until capacity saved by manual deletion) 1: Deletes preferentially from past watched programme 2: Deletes preferentially from old programme (follow setting of getPVRSaveDays(), getPVRSaveEpisodes())
Returns	None
Description	Sets preferential policy for storage and deletion of recorded programme
int **getPVRSaveDays**()	
Parameters	None
Returns	Storage period (day) of recorded programme
Description	Gets default value of minimum days for storage of recorded programme by system
void **setPVRSaveDays**(int days)	
Parameters	days: Storage period (day) of recorded programme
Returns	None
Description	Sets default value of minimum days for storage of recorded programme by system
int **getPVRSaveEpisodes**()	
Parameters	None
Returns	Number of stored episodes in same series
Description	sets default value of minimum episodes number to be stored by system for same series of recorded programme
void **setPVRSaveEpisodes**(int episodes)	
Parameters	episodes: Stored episodes number
Returns	None
Description	Sets default value of minimum number of episodes to be stored by system for same series of recorded programme

Table III.56 – Method of SystemConfig Class

int **getPVRStartPadding**()	
Parameters	None
Returns	Default value (second) of start padding time of record start time
Description	Gets default value in seconds for start padding before designated recording start time such as programme start time
void **setPVRStartPadding** (int padding)	
Parameters	padding: Default value (second) for start padding time before recording start time
Returns	None
Description	Sets default value in second for start padding before designated recording start time such as programme start time
int **getPVREndPadding**()	
Parameters	None
Returns	Default value (second) of end padding time of record end time
Description	Gets default value in second for end padding after designated recording end time such as programme end time
void **setPVREndPadding** (int padding)	
Parameters	padding: Default value (second) of end padding time after recording end time
Returns	None
Description	Sets default value in second for end padding after designated recording end time such as programme end time
boolean **isAutoPPVPurchaseEnabled**()	
Parameters	None
Returns	True if automatic PPV purchase mode is effective
Description	Confirms whether automatic PPV purchase mode is effective or not
void **enableAutoPPVPurchase**(boolean flag)	
Parameters	flag: True in case of effecting automatic PPV purchase mode
Returns	None
Description	Sets whether STB automatic PPV purchase mode to be effective or not
PPVHistory[] **getPPVHistory**()	
Parameters	None
Returns	Matrix of PPV purchase history data
Description	Gets PPV purchase history data of STB
int **getTotalPPVPurchasedFee**()	
Parameters	None
Returns	Total fee by PPV purchase history
Description	Gets total fee based on PPV purchase history of STB
boolean **uploadPPVViewLogData**()	
Parameters	None
Returns	True if it is successful
Description	Uploads PPV watching history data of STB

Table III.56 – Method of SystemConfig Class

void **deleteAllPPVPrivateData**()	
Parameters	None
Returns	None
Description	Deletes PPV private data of STB
ChannelScanParameter[] **getSystemChannelScanParameters**()	
Parameters	None
Returns	Matrix of system set value for channel scan parameter
Description	Gets system set value for channel scan parameter stored in STB
java.lang.String **getBCASCardId**()	
Parameters	None
Returns	Character set of B-CAS card ID
Description	Gets character set of CAS card ID inserted in B-CAS card slot
java.lang.String **getCCASCardId**()	
Parameters	None
Returns	Character set of C-CAS card ID
Description	Gets character set of CAS card ID inserted in C-CAS card slot
java.lang.String **getBCASCardStatus**()	
Parameters	None
Returns	Character set designates B-CAS card status ("OK" or "NG")
Description	Confirms CAS status inserted in B-CAS card slot of STB
java.lang.String **getCCASCardStatus**()	
Parameters	None
Returns	Character set designates C-CAS card status ("OK" or "NG")
Description	Confirms CAS status inserted in C-CAS card slot of STB
java.util.Date **getSystemUpdateDate**()	
Parameters	None
Returns	Day and time of final system update
Description	Gets day and time of final system update of STB
int **getAvailableInputEvent**()	
Parameters	None
Returns	Descriptor for remote controller key event (Table III.54)
Description	Gets available method by bitflag as a getting method of remote controller key event. If RCTYPE_CALLBACK bit is 1, using onControllerKeyUp() and onControllerKeyDown() of SystemEventListener remote controller key event can be obtained. If RCTYPE_KEYBOARD bit is 1, remote controller key event can be obtained by getting keyboard input event and key mapping by getEmulatedKeyCode() method.
boolean **isKeyboardMode**()	
Parameters	None
Returns	True in case of Android keyboard input mode
Description	Confirms whether getting method of remote controller key event is set to keyboard input mode.

Table III.56 – Method of SystemConfig Class

boolean **setKeyboardMode**(boolean flag)	
Parameters	flag: True in case of keyboard input mode
Returns	True if input mode change is successful
Description	Changes getting method of remote controller key event. Setting true in parameters means keyboard input mode is set, setting false, event listener input mode is set. However if other method applies except for mode which can be set by isKeyboardMode(), returns become false and it cannot be set.
int **getEmulatedKeyCode**(int keyCode)	
Parameters	keyCode: key
Returns	Label name of related remote controller key (Table III.55)
Description	Gets label name of remote controller key related to key event
boolean **setEmulatedKeyCode**(int rcKeyCode, int keyCode)	
Parameters	rcKeyCode: Label name of remote controller key (Table III.55) keyCode: Android key to be related (android.view.KeyEvent.KEYCODE_*)
Returns	True if change is successful
Description	Sets key event related to label name of remote controller key
void **setSystemEventListener**(SystemEventListener listenerClass)	
Parameters	listenerClass: Class to be registered as event listener
Returns	None
Description	Registers class to be designated by parameter as event listener to be called when event related to STB system occurs

III.4.3 SystemMessage Class

java.lang.Object

└java-cable.or.int.config.SystemMessage

This is a class for the provisioning of the read-write function of the system message.

III.4.3.1 Definition of class in class

Constructors related to this interface are defined in Table III.57.

Table III.57 – Contractor of SystemMessage.EmmMessage class

SystemMessage.EmmMessage()	
Parameters	None
Description	Generates class instance after initialization
SystemMessage.EmmMessage(int id, int type, int networkType, java.util.Date date, java.lang.String body)	
Parameters	id: Message ID type: Kind of message (0: EMM, 1: internal) networkType: Descriptor of broadcast network (Table III.2) date: date of message body: body of message
Description	Generates class instance after initialization by indicating message parameter and content specifically

Methods related to this interface are defined in Table III.58.

Table III.58 – Method of SystemMessage.EmmMessage class

int **getId**()	
Parameters	None
Returns	Message ID
Description	Gets EMM message ID value
int **getType**()	
Parameters	None
Returns	Kind of message (0: EMM, 1: internal)
Description	Confirms kind of EMM message
int **getNetworkType**()	
Parameters	None
Returns	Descriptor of broadcast network (Table III.2)
Description	Confirms broadcast network where EMM message generated
java.util.Date **getDate**()	
Parameters	None
Returns	Date of message
Description	Confirms date of EMM message is generated
java.lang.String **getBody**()	
Parameters	None
Returns	Body of message
Description	Confirms body of EMM message
boolean **isRead**()	
Parameters	None
Returns	True if body of message is confirmed
Description	Gets information whether body of EMM message is confirmed

Constructors related to this interface are defined in Table III.59.

Table III.59 – Contractor of SystemMessage.BoardMessageClass

SystemMessage.BoardMessage()	
Parameters	None
Description	Generates class instance after initialization
SystemMessage.BoardMessage(int networkType, java.lang.String body)	
Parameters	networkType: Descriptor of broadcast network (Table III.2) body: body of message
Description	Generates class instance after initialization by indicating message parameter and content specifically

Methods related to this interface are defined in Table III.60.

Table III.60 – Method of SystemMessage.BoardMessageClass

int **getNetworkType**()	
Parameters	None
Returns	Descriptor of broadcast network (Table III.2)
Description	Confirms broadcast network where EMM message is generated
java.lang.String **getBody**()	
Parameters	None
Returns	Body of message
Description	Confirms body of EMM message

III.4.3.2 Method

Methods related to this interface are defined in Table III.61.

Table III.61 – Method of SystemMessage class

SystemMessage.EmmMessage[] **getEmmMessages**()	
Parameters	None
Returns	String of EMM message
Description	Gets EMM message (value of getType()is 0)
SystemMessage.EmmMessage **getEmmMessageById**(int id)	
Parameters	id: Message ID
Returns	EMM message linked with ID
Description	Gets EMM message linked with message ID
boolean **deleteEmmMessage**(int id)	
Parameters	id: Message ID
Returns	True if deletion is successful
Description	Deletes EMM message linked with message ID
boolean **setEmmMessageRead**(int id)	
Parameters	id: Message ID
Returns	True if change is successful
Description	Sets EMM message linked with message ID in confirmed status
SystemMessage.EmmMessage[] **getSelfMessages**()	
Parameters	None
Returns	String of internal message
Description	Gets STB internal message (value of getType()is 1)
boolean **deleteSelfMessage**(int id)	
Parameters	id: Message ID
Returns	True if deletion is successful
Description	Deletes internal message linked with message ID
boolean **setSelfMessageRead**(int id)	
Parameters	id: Message ID
Returns	True if change is successful
Description	Sets internal message linked with message ID in confirmed status

Table III.61 – Method of SystemMessage class

SystemMessage.BoardMessage[] **getBoardMessages**()	
Parameters	None
Returns	String of board message
Description	Gets board message
void **stopAutoDisplayMessagey**(int id)	
Parameters	id: EMM massage ID
Returns	None
Description	Stops automatic display of EMM message linked with ID

III.4.4 ChannelScanParameter Interface

java-cable.or.int.config.param.ChannelScanParameter

Interface to provide definition of channel scan parameter.

III.4.4.1 Method

Methods related to this interface are defined in Table III.62.

Table III.62 – Method of ChannelScanParameter class

int **getStartFrequency**()	
Parameters	None
Returns	Channel scan start frequency
Description	Gets channel scan start frequency
void **setStartFrequency**(int frequency)	
Parameters	frequency: Channel scan start frequency
Returns	None
Description	Sets channel scan start frequency
int **getEndFrequency**()	
Parameters	None
Returns	Channel scan end frequency
Description	Gets channel end frequency
void **setEndFrequency**(int frequency)	
Parameters	frequency: Channel scan end frequency
Returns	None
Description	Sets channel scan end frequency
int **getModulationModes**()	
Parameters	None

Table III.62 – Method of ChannelScanParameter class

Returns	Bit flag of related modulation mode (can be plural by OR) 0x01: QPSK 0x02: 8PSK 0x04: 16QAM 0x08: 32QAM 0x10: 64QAM 0x20: 128QAM 0x40: 256QAM
Description	Gets related modulation mode of channel scan
void **setModulationModes**(int modes)	
Parameters	modes: Bit flag of related modulation mode (can be plural by OR) 0x01: QPSK 0x02: 8PSK 0x04: 16QAM 0x08: 32QAM 0x10: 64QAM 0x20: 128QAM 0x40: 256QAM
Returns	None
Description	Sets related modulation mode of channel scan
int **getNetworkType**()	
Parameters	None
Returns	Descriptor of broadcast network (Table III.2)
Description	Confirms broadcast network to be scanned
java.lang.String **getFavId**()	
Parameters	None
Returns	Favourite list ID
Description	Gets favourite list ID which links channel list to be detected by channel scan
boolean **setFavId**(java.lang.String favId)	
Parameters	favId: Favourite list ID
Returns	True if change is successful
Description	Designate and sets channel list to be detected by channel scan using favourite list ID
void **restoreSystemSettings**(int networkType)	
Parameters	networkType: Descriptor of broadcast network (Table III.2)
Returns	None
Description	Initialize to channel scan parameter which set by system

III.4.5 DVBTChannelScanParameter Class

java.lang.Object

└**java-cable.or.int.config.param.DVBTChannelScanParameter**
(implements ChannelScanParameter)

This class is to provide the channel scan parameter of a terrestrial digital broadcast. The only difference with ChannelScanParameter is described below.

III.4.5.1 Method

Methods related to this interface are defined in Table III.63.

Table III.63 – Method of DVBTChannelScanParameter class

int **getRaster**()	
Parameters	None
Returns	Raster size (kHz)
Description	Gets raster size of channel scan
void **setRaster**(int size)	
Parameters	size: raster size
Returns	None
Description	Sets raster size of channel scan
java.lang.String **getOfdm**()	
Parameters	None
Returns	OFDM Mode "MODE_1K": OFDM Mode 1K "MODE_2K": OFDM Mode 2K "MODE_4K": OFDM Mode 4K "MODE_8K": OFDM Mode 8K "MODE_16K": OFDM Mode 16K "MODE_32K": OFDM Mode 32K
Description	Gets OFDM mode of channel scan
void **setOfdm**(java.lang.String mode)	
Parameters	mode: OFDM Mode "MODE_1K": OFDM Mode 1K "MODE_2K": OFDM Mode 2K "MODE_4K": OFDM Mode 4K "MODE_8K": OFDM Mode 8K "MODE_16K": OFDM Mode 16K "MODE_32K": OFDM Mode 32K
Returns	None
Description	Sets OFDM mode of channel scan
java.lang.String **getBandwidth**()	
Parameters	None
Returns	Frequency band mode "BAND_1.7MHz": 1.7MHz "BAND_5MHz": 5MHz "BAND_6MHz": 6MHz "BAND_7MHz": 7MHz "BAND_8MHz": 8MHz "BAND_10MHz": 10MHz

Table III.63 – Method of DVBTChannelScanParameter class

Description	Gets frequency band of channel scan
void **setBandwidth**(java.lang.String mode)	
Parameters	mode: Frequency band mode "BAND_1.7MHz": 1.7MHz "BAND_5MHz": 5MHz "BAND_6MHz": 6MHz "BAND_7MHz": 7MHz "BAND_8MHz": 8MHz "BAND_10MHz": 10MHz
Returns	None
Description	Sets frequency band of channel scan

III.4.6 DVBSChannelScanParameterClass

java.lang.Object

└**java-cable.or.int.config.param.DVBSChannelScanParameter**
(implements ChannelScanParameter)

This class is to provide the channel scan parameter of a BS digital broadcast. The only difference with ChannelScanParameter is described below.

III.4.6.1 Method

Methods related to this interface are defined in Table III.64.

Table III.64 – Method of DVBSChannelScanParameter class

java.lang.String **getSymbolRate**()	
Parameters	None
Returns	Assigns plural symbol rate segregated with comma
Description	Gets symbol rate of channel scan
void **setSymbolRate**(java.lang.String rate)	
Parameters	rate: Plural symbol rate segregated with comma
Returns	None
Description	Sets symbol rate of channel scan
int **getPolarization**()	
Parameters	None
Returns	Bit flag of polarization mode (can be plural by OR) 0x01: Horizontal polarization 0x02: Vertical polarization 0x04: Clockwise circular polarization 0x08: Counter clockwise circular polarization
Description	Gets polarization of channel scan

Table III.64 – Method of DVBSChannelScanParameter class

void **setPolarization**(int mode)	
Parameters	mode: Bit flag of polarization mode (can be plural by OR) 0x01: Horizontal polarization 0x02: Vertical polarization 0x04: Clockwise circular polarization 0x08: Counter clockwise circular polarization
Returns	None
Description	Sets polarization of channel scan
java.lang.String **getCodeRate**()	
Parameters	None
Returns	Character string for code rate (ex. "3/4", "5/6")
Description	Gets code rate of channel scan
void **setCodeRate**(java.lang.String codeRate)	
Parameters	codeRate: Character string for code rate (ex. "3/4", "5/6")
Returns	None
Description	Sets code rate of channel scan
double **getOrbitalPosition**()	
Parameters	None
Returns	orbital position
Description	Gets setting value of orbital position
void **setOrbitalPosition**(double position)	
Parameters	position: orbital position
Returns	None
Description	Sets orbital position
int **getNetworkId**()	
Parameters	None
Returns	networkId: Network ID (If minus value, all network ID is targeted)
Description	Gets set value of network ID which is target of channel scan
void **setNetworkId**(int networkId)	
Parameters	networkId: Network ID (If minus value, all network ID is targeted)
Returns	None
Description	Sets network ID which is target of channel scan

III.4.7 DVBCChannelScanParameter Class

java.lang.Object

└**java-cable.or.int.config.param.DVBCChannelScanParameter**
(implements ChannelScanParameter)

This class is to provide the channel scan parameter of a CATV digital broadcast. The only difference with ChannelScanParameter is described below.

III.4.7.1 Method

Methods related to this interface are defined in Table III.65.

Table III.65 – Method of DVBCChannelScanParameter Interface

int **getRaster**()	
Parameters	None
Returns	Raster size (kHz)
Description	Gets raster size of channel scan
void **setRaster**(int size)	
Parameters	size: raster size
Returns	None
Description	Sets raster size of channel scan
boolean **isStartNetworkScanOnNIT**()	
Parameters	None
Returns	True, if it is only scanning of frequency band given by NIT
Description	Confirms whether to limit scanning to only frequency band given by NIT
void **setStartNetworkScanOnNIT**(boolean flag)	
Parameters	flag: True, if it is only scanning of frequency band given by NIT
Returns	None
Description	Sets whether to limit scanning to only frequency band given by NIT
java.lang.String **getSymbolRate**()	
Parameters	None
Returns	Assigns plural symbol rate segregated with comma
Description	Gets symbol rate of channel scan
void **setSymbolRate**(java.lang.String rate)	
Parameters	rate: Assigns plural symbol rate segregated with comma
Returns	None
Description	Sets symbol rate of channel scan
int **getNetworkId**()	
Parameters	None
Returns	Network ID (If minus value, all network ID is targeted)
Description	Gets network ID setting value to be scanned
void **setNetworkId**(int networkId)	
Parameters	networkId: Network ID (If minus value, all network ID is targeted)
Returns	None
Description	Sets network ID which is target of channel scan

III.4.8 ParentalControlClass

java.lang.Object

└java-cable.or.int.config.param.ParentalControl

This is a class of control for setting of parental control.

III.4.8.1 Field

Identifiers related to this interface are defined in Table III.66.

Table III.66 – PIN Block Descriptor

Name of Field	Value	Content
PIN_CORRECT	0	Unlock PIN by correct PIN code
PIN_CORRECT	1	Input of wrong PIN code
PIN_CORRECT	2	PIN lock
INVALID_OBJECT	3	Wrong object

III.4.8.2 Method

Methods related to this interface are defined in Table III.67.

Table III.67 – Method of ParentalControl Class

ParentalRatingScheme[] **getParentalRatingSchemes**()	
Parameters	None
Returns	String of parental rating scheme information
Description	Gets parental rating scheme information (scheme name and threshold for scheme), See clause C.3.2.10 in detail
ParentalRatingScheme **addParentalRatingScheme**(java.lang.String name, java.lang.String values)	
Parameters	name: Scheme name (equivalent to ParentalRating#getScheme()) values: threshold value
Returns	Added parental rating scheme information
Description	In parental rating, sets threshold value of parental rating additionally for assigned scheme
ParentalRatingScheme **getParentalRatingScheme**(java.lang.String name)	
Parameters	name: Scheme name (equivalent to ParentalRating#getScheme())
Returns	Applicable information of parental rating scheme
Description	Gets threshold value of parental rating for assigned scheme
boolean **isPINEntryLocked**()	
Parameters	None
Returns	True, if input of PIN code is locked
Description	Confirms whether it is locked for not enabling PIN code input against for multiple access of wrong PIN code input
boolean **getParentalControlStatus**()	
Parameters	None
Returns	True, if rating block by parental control works
Description	Confirms whether rating block by parental control is enabled and works by user setting, etc.
int **setParentalControlStatus**(java.lang.String pcPIN, boolean enable)	
Parameters	pcPIN: PIN code enable: True if it enables parental control
Returns	Result of PIN code input (Table III.66)

Table III.67 – Method of ParentalControl Class

Description	Changes enable/unenable status of parental control. Change of setting is limited to the case of right PIN code input only, and then accepted. If the PIN code is wrong, an error code is sent and the change is rejected.
boolean **getBlockUnrated**()	
Parameters	None
Returns	True, if rating block is applied to the programme which has no parental rating setting
Description	Confirms whether rating block is enabled in the programme which has no parental rating limit by user setting, etc.
int **setBlockUnrated**(java.lang.String pcPIN, boolean blocked)	
Parameters	pcPIN: PIN code enable: True, if it enables rating block to the programme which has no parental rating limit.
Returns	Result to PIN code input (Table III.66)
Description	Changes enable/unenable status of parental control in the programme which has no parental rating limit. Change of setting is limited to the case of right PIN code input only, and then accepted. If the PIN code is wrong, an error code is sent and the change is rejected.
int **verifyParentalControlPIN**(java.lang.String pcPIN)	
Parameters	pcPIN: PIN code
Returns	Result to PIN code input (Table III.66)
Description	Confirms whether PIN code is right or not
int **setParentalControlPIN**(java.lang.String oldPcPIN, java.lang.String newPcPIN)	
Parameters	oldPcPIN: PIN code before change newPcPIN: PIN code after change
Returns	Result to PIN code input before change (Table III.66)
Description	Changes PIN code. PIN code after change is applied to STB in the case of right PIN code input only. If the PIN code before change is wrong, an error code is sent and the change is rejected.
int unlockWithParentalControlPIN(java.lang.String pcPIN, java.lang.Object target)	
Parameters	PcPIN: PIN code object: Object to be applied to PIN lock release
Returns	Result to PIN code input and Object application (Table III.66)
Description	Releases parental rating limit to specific object (Channel, Player). Change of setting is limited to the case of right PIN code input only, and then accepted. If the PIN code is wrong, an error code is sent and the change is rejected.

III.4.9 ParentalRating Class

java.lang.Object

└java-cable.or.int.config.param.ParentalRating

This is a class for parental rating limit information of a programme and channel. It shows a standard value of rating limit indicated by programme or channel. If the limit indicated by the ParentalRatingScheme set by the user is in conflict with the limit rate of ParentalRating, watching the programme or channel is to be blocked.

Constructors related to this interface are defined in Table III.68.

Table III.68 – Constructor of ParentalRating Class

ParentalRating(java.lang.String name, java.lang.String scheme, int value, int labels, java.lang.String region)	
Parameters	name: Setting name of parental rating (ex. PG-13) scheme: Scheme name value: Limit value labels: Limit label region: Region name
Description	Generates class instance with given content after initialization

III.4.9.1 Method

Methods related to this interface are defined in the following table.

Table III.69 – Method of ParentalRating Class

java.lang.String **getName**()	
Parameters	None
Returns	Name of parental rating
Description	Gets name of parental rating
java.lang.String **getScheme**()	
Parameters	None
Returns	Scheme name of parental rating
Description	Gets scheme name of parental rating
int **getValue**()	
Parameters	None
Returns	Limit value of parental rating
Description	Gets limit value of parental rating
int **getLabels**()	
Parameters	None
Returns	Bit flag of limit label of parental rating (can be assigned multiple by "OR") 0x0001: Including scene suggesting sexual expression 0x0002: Including exaggerated language expression 0x0004: Including sexual expression 0x0008: Including violence expression 0x0010: Including non-realistic violence expression 0x0020: Including discomfort visual expression 0x0040: Including crime related expression 0x0080: Including illegal drug related expression 0x0100: Including threatening expression which may cause cramp or unconsciousness
Description	Gets limit label of parental rating
java.lang.String **getRegion**()	
Parameters	None
Returns	Region of parental rating
Description	Gets region of parental rating

III.4.10 ParentalRatingSchemeClass

ParentalRatingScheme Class

java.lang.Object

└java-cable.or.int.config.param.ParentalRatingScheme

This is a class dealing with parental audio visual control schemes that have been set by the user.

III.4.10.1 Method

Methods related to this interface are defined in Table III.70.

Table III.70 – Method of ParentalRatingScheme Class

java.lang.String **getName**()	
Parameters	None
Returns	Parental audio visual control schemes name
Description	Gets parental audio visual control schemes name
int **getThreshold**()	
Parameters	None
Returns	Threshold for parental audio visual control schemes
Description	Gets threshold for parental audio visual control schemes
Android.net.Uri **getIconUri**()	
Parameters	None
Returns	URI of the icon
Description	To get the URL of the icon for the ABC. (If present)

III.4.11 PPVHistoryClass

PPVHistory Class

java.lang.Object

└java-cable.or.int.config.param.PPVHistory

This is a class dealing with parental audio visual control schemes that have been set by the user

III.4.11.1 Method

Methods related to this interface are defined in Table III.71.

Table III.71 – Method of PPVHistory Class

java.util.Date **getDate**()	
Parameters	None
Returns	PPV purchase date and time
Description	Gets PPV purchase date and time
void **setDate**(java.util.Date date)	
Parameters	date: PPV purchase date and time
Returns	None
Description	Sets the PPV purchase date and time

Table III.71 – Method of PPVHistory Class

int **getFee**()	
Parameters	None
Returns	PPV purchase price
Description	Gets PPV purchase price
void **setFee**(int fee)	
Parameters	fee: PPV purchase price
Returns	None
Description	Sets PPV purchase price
java.lang.String **getEventData**()	
Parameters	None
Returns	Meta information concerning PPV purchase
Description	Gets meta information concerning PPV purchase
void **setEventData**(java.lang.String data)	
Parameters	date: Meta information concerning PPV purchase
Returns	None
Description	Sets meta information concerning PPV purchase
java.lang.String **getCaSystemId**()	
Parameters	None
Returns	Refer to CAS Identifier concerning PPV purchase (PlayerEventListener#onParentalRatingError()
Description	Gets identifier of the CAS concerning PPV purchase

III.4.12 SystemEventListenerInterface

SystemEventListener interface

java-cable.or.int.config.event.SystemEventListener

defines the methods of the event listener which receives the event when an event related to the overall system has occurred.

III.4.12.1 Method

Methods related to this interface are defined in Table III.72.

Table III.72 – Method of SystemEventListener interface

void **onEmmAutoDisplayMessageNotify**(int id, int autoDisplayType)	
Parameters	id: EMM message ID autoDisplayType: Is as follows 0: Can be deleted 1: Cannot be deleted 2: Delete a message from the display screen
Description	Notifies automatically display EMM message ID
void **onEmmNewMessageNotify**(int type, int id);	
Parameters	type: Message type(0: EMM, 1: Internal) id: Message ID
Description	Notifies addition of EMM messages and internal messages
void **onControllerKeyDown**(int keyCode)	
Parameters	keyCode: Descriptors for Remote Controller Key Event (Table III.54)
Description	notifies remote control key push
void **onControllerKeyUp**(int keyCode)	
Parameters	keyCode: Descriptors for Remote Controller Key Event (Table III.54)
Description	Notifies that a pushed remote controller key was not pushed

III.4.13 Broadcast Intent

Permissible events as broadcast intent are as follows.

Broadcast intent names and label names used in this interface are defined in Tables III.73 and III.74.

Table III.73 – List of Broadcast intent

IntentName	Description
java-cable.or.int.intent.action. EMM_AUTO_DISPLAY_MESSAGE_NOTIFY	Notifies automatically display EMM message ID Equivalent to SystemEventListener#onEmmAutoDisplayMessageNotify()
java-cable.or.int.intent.action. EMM_NEW_MESSAGE_NOTIFY	Notifies addition of EMM messages and internal messages Equivalent to SystemEventListener#onEmmNewMessageNotify()

The variable name of the parameter (the first argument of getExtra) can be obtained by getExtra at the time of reception of the above-mentioned broadcasting Intent as a field value of a SystemIntent class.

Table III.74 – Variable name label of broadcast Intent

Label name (Field of SystemIntent)	Type	Description
EMM_ID	Integer	Equivalent to the 1st argument of SystemEventListener#onEmmAutoDisplayMessageNotify() Equivalent to the 2nd argument of SystemEventListener#onEmmNewMessageNotify()
EMM_AUTO	Integer	Equivalent to the 2nd argument of SystemEventListener#onEmmAutoDisplayMessageNotify()
EMM_TYPE	Integer	Equivalent to the 1st argument of SystemEventListener#onEmmNewMessageNotify()

Appendix IV

DAE API Extensions

(This appendix does not form an integral part of this Recommendation.)

This specification is intended to define the declarative application environment (DAE) based on [b-OIPF DAE] Open IPTV Forum (OIPF) RELEASE 2 Specification Declarative Application.

[b-OIPF DAE] partially supports the DVB network, but it was not designed to support other broadcast networks in other regions, including the Japanese Terrestrial, BS, and CATV networks. Therefore, this document tries to use the original parts of [b-OIPF DAE] as much as possible and tries to extend the specification to support the Japanese broadcast network and JLABs specification. In order to satisfy items in Table IV.1, this document describes API extended from [b-OIPF DAE] and newly extended API. Description of standard API of [b-OIPF DAE] is out of the scope of this Recommendation. The "Cable" notation in Appendix IV is an example of name space given in this Recommendation. Other names can be applicable for actual implementation. To maintain the interoperability of the application as much as possible, it is very useful to provide information on change in the name space by implementer.

IV.1 General

The overall architecture of the declarative application environment is shown in Figure IV.1.

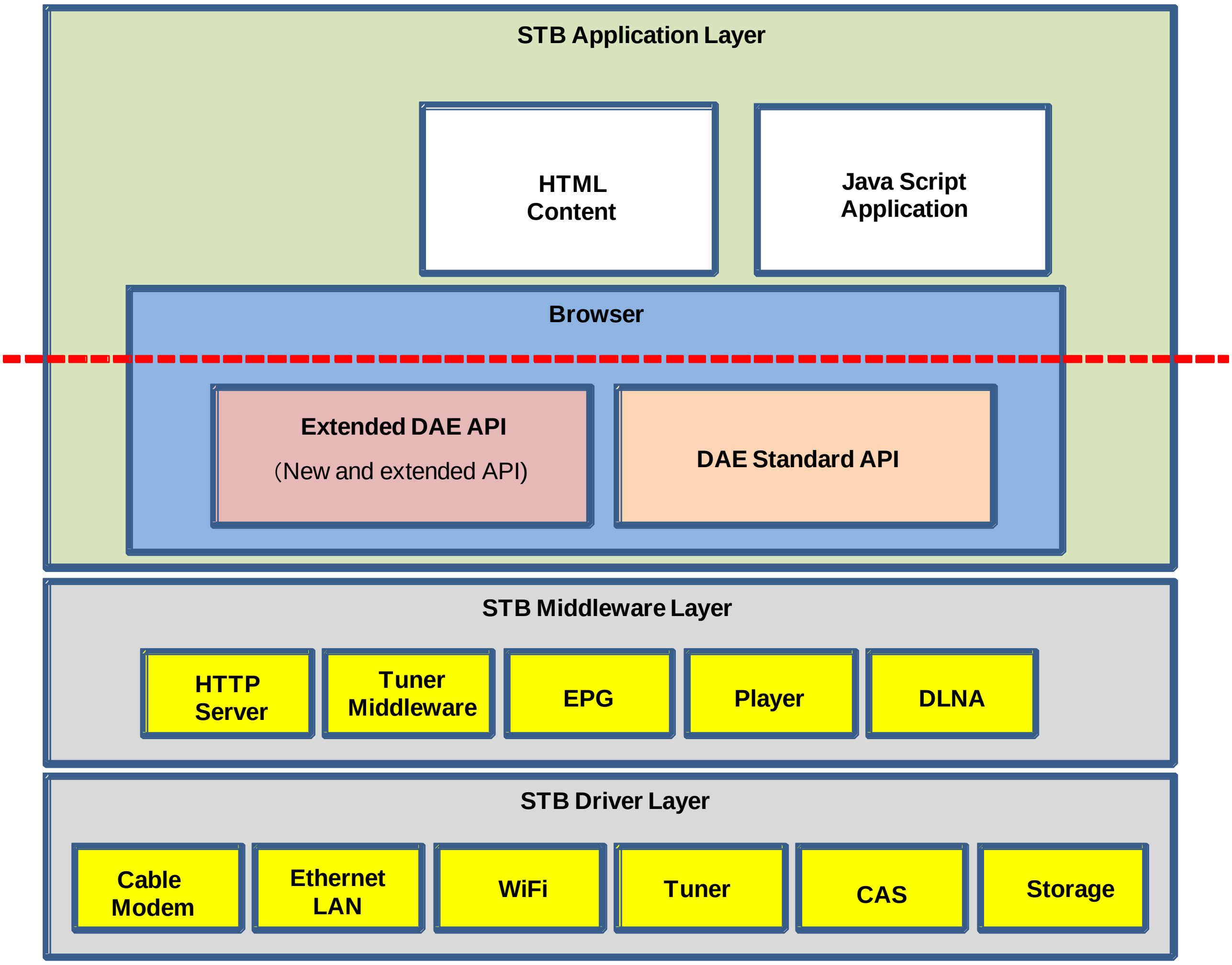


Figure IV.1 – Overall DAE architecture

Functions to be provided by the declarative application environment are classified into four major blocks.

1. Tune to ARIB terrestrial/broadcast satellite/Japanese cable TV networks.
2. Display EPG grid screen and indicate event programme information.
3. Execute the detailed STB setting.
4. Offer PVR/home network functionality.

However, [b-OIPF DAE] clause 7, defines classes (see Table IV.1) and cannot fully support ARIB or CATV networks.

Table IV.1 – [b-OIPF DAE] Class

High-level Function (which can include several embedded objects)	Description
Application Manager	Overall application management, behaviour ,and tasking such as support for multiple simultaneous applications, inter-application communication and application signalling
Capabilities	Access to the terminal's capability description
Configuration	Access to device configuration and user settings
Remote Management	Remote management for accessing device diagnosis and software upgrade
DRM	Acquisition of content right of protected content and unification of video and audio object for the communication with content protection system
CoD Maneger	Same as access to on-demand content catalogue, display control of unicast media, recording media and download media including trick media
Download Manager	Same as media download queue and download item, basic guide of media download of protected/non-protected content
Scheduled Content	Broadcast video display control including trick play recording and synchronized application to video
Parental Control Manager	Control of parental control function in access control , reception and PIN control
Channel List Management	Discovery and control of channel list and favourite list, including channel scanning
Recording Control	Storage and search of local and network recording schedule and book mark
Metadata Control	Search support of programme guide and VoD content catalogue

This clause defines new API to [b-OIPF DAE] in order to extend functions which lack in [b-OIPF DAE] standard

Overall DAE components are shown in Figure IV.2.

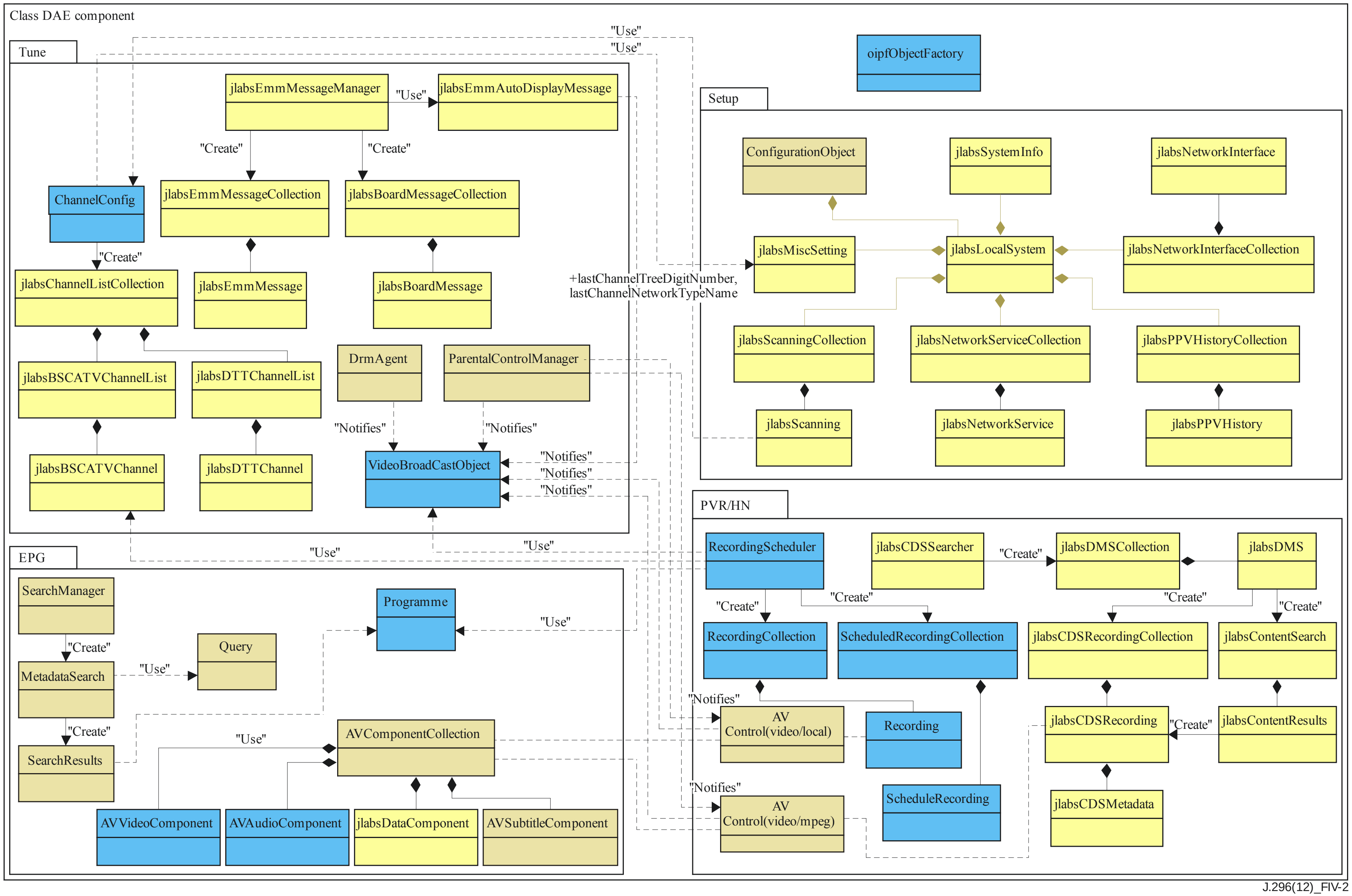


Figure IV.2 – Overall component diagram

This clause describes the overall component diagram that cable DAE API intends to define. The components consist of four blocks.

1. Tune
2. EPG
3. Setting
4. PVR/HN

An overall component diagram is shown in Figure IV.2.

The yellow class in the component diagram indicates the cable new class to [b-OIPF DAE] and the blue class indicates the extension to standard [b-OIPF DAE].

IV.1.1 Tune

New classes are defined to support ARIB/CATV networks. This is because Japanese DTT/BS/CATV network has a three digit/one touch channel selection use case and is different from the original [b-OIPF DAE] APIs. [b-OIPF DAE] does not define EMM/board message handling classes and defines new classes.

IV.1.2 EPG

Digital/analogue copy control information is lacking in the programme class and AVVideoComponent and AVAudioComponent class need to have more detailed information about "content_type" and "stream_type". The component class needs DATA type component and adds a new class.

IV.1.3 Setup

[b-OIPF DAE] does not define any detailed setting API in Configuration and Setting APIs, so Cable DAE API defines the new classes. The new classes consists of "cableNetworkInterface", "cableNetworkService", "cablePPVHistory", "cableSystemInfo", "cableScanning" and "cableMiscSetting".

IV.1.4 PVR/HN

For the cae of PVR use, [b-OIPF DAE] APIs are applicable, however, there should be one additional method to retrieve the "Recording" class. For the home network extension, there are new classes of "cableCDSSearcher", "cableCDS", "cableCDSRecording", "cableCDSMetadata", "cableContentSearch", "cableContentResults" and related collection classes.

IV.1.5 Class

Table IV.2 shows the cable extended API class.

Table IV.2 – Cable extended API class

No.	Category	Name of Class	Summary
General			
1	Extension to current Spec	oipfObjectFactory	Addition of method to create cable new class
Tune			
2	Extension to current Spec	ChannelConfig	Extension to access cableDTTChannel/ cableBSCATVChannel (Cable new class)

Table IV.2 – Cable extended API class

No.	Category	Name of Class	Summary
3	Extension to current Spec	Video/Broadcast	Extension of [b-OIPF DAE] Standard to meet ARIB/CATV standard
4	New extension	cableChannelListCollection	Collection of cableDTTChannelList/cableBSCATVChannelList
5	New extension	cableDTTChannelList	Getting channel information list for terrestrial digital broadcast
6	New extension	cableDTTChannel	Getting channel information for terrestrial digital broadcast
7	New extension	cableBSCATVChannelList	Getting channel information list for BS digital broadcast
8	New extension	cableBSCATVChannel	Getting channel information for BS digital broadcast
9	New extension	cableEmmMessageManager	Control of EMM, internal mail and board message
10	New extension	cableEmmAutoDisplayMessage	Display of automatic display message
11	New extension	cableEmmMessage	Display of EMM message
12	New extension	cableEmmMessageCollection	Collection of cableEmmMessage
13	New extension	cableBoardMessage	Display of board message
14	New extension	cableBoardMessageCollection	Collection of cableBoardMessage
EPG			
15	Extension to current Spec	Programme	Extension of [b-OIPF DAE] standard programme class to meet ARIB/CATV specification
16	Extension to current Spec	AVVideoComponent	Extension of [b-OIPF DAE] AVVideoComponent class to meet ARIB/CATV specification
17	Extension to current Spec	AVAudioComponent	Extension of [b-OIPF DAE] AVAudioComponent class to meet ARIB/CATV specification
18	Extension to current Spec	Media Playback	Extension of AV Control object class defined in CEA-2014 that [b-OIPF DAE] refers to meet ARIB/CATV specification
19	New extension	cableDataContentComponent	Extension of data component class to meet BML specification
Setup			
20	New extension	cableNetworkInterface	Application of [b-OIPF DAE] standard NetworkInterface class to IPv6
21	New	cableNetworkInterfaceCollection	Collection of cableNetworkInterface

Table IV.2 – Cable extended API class

No.	Category	Name of Class	Summary
	extension		
22	New extension	cableNetworkService	Extension of setting for wireless LAN, WAN, WiFi and DHCP server, etc.
23	New extension	cableNetworkServiceCollection	Collection class of cableNetworkService
24	New extension	cableMiscSetting	Setting except for network, scan, PPV and parental control within Cable extension
25	New extension	cableSystemInfo	Setup class defining STB fixed value
26	New extension	cablePPVHistory	Getting PPV history
27	New extension	cablePPVHistoryCollection	Collection of cablePPVHistory
28	New extension	cableLocalSystem	Collection of network interface service, STB setting information and PPV history
29	New extension	cableScanning	Setting of Channel Scanning
30	New extension	cableScanningCollection	Collection of cableScanning
PVR/HN			
31	Extension to current Spec	RecordingScheduler	Extension of [b-OIPF DAE] standard RecordingScheduler class to meet ARIB/CATV specification
32	Extension to current Spec	Recording	Extension of [b-OIPF DAE] standard Recording class to meet ARIB/CATV specification
33	Extension to current Spec	ScheduledRecording	Extension of [b-OIPF DAE] standard ScheduledRecording class to meet DLNA recording reservation specification
34	New extension	cableCDSSearcher	Search of DLNA DMS in home network
35	New extension	cableDMSCollection	Collection of cableDMS
36	New extension	cableDMS	Designation of cableDMS
37	New extension	cableContentSearch	Search of recorded content over DMS
38	New extension	cableContentResults	Getting search result of recorded content over DMS
39	New extension	cableCDSRecordingCollection	Collection class of cableCDSRecording
40	New extension	cableCDSRecording	Designation of recorded content over DLNA DMS
41	New extension	cableCDSMetadata	Designation of optional property metadata over DLNA DMS

IV.1.6 API

This clause describes [b-OIPF DAE] and cable rxtended DAE API as the declarative API. The browser-specific API is out of the scope of this Recommendation. This Recommendation describes details of the cable extended API in a component basis.

IV.2 Relation to overall system

IV.2.1 Cable extension to [b-OIPF DAE]

Cable extensions to oipfObjectFactory

IV.2.1.1 Addition of method to create Cable new class

(1) Method (Extension to [b-OIPF DAE] clause 7.1)

Methods related to this API are defined in Table IV.3.

Table IV.3 – Additional Method to oipfObjectFactory

cableEmmMessageManager **createCableEmmMessageManager**()	
Parameters	None
Return	jlabEmmMessageManager
Description	If the object type is supported, this method SHALL return an instance of the corresponding cableEmmMessageManager object. Supported MIME type is application/oipfCableEmmMessageManager.
cableCDSSearcher **createCableCDSSearcher** ()	
Parameters	None
Return	jlabCDSSearcher
Description	If the object type is supported, this method SHALL return an instance of the corresponding cableCDSSearcher object. Supported MIME type is "application/oipfCableCDSSearcher"

IV.3 Broadcast Reception

IV.3.1 Cable extension to [b-OIPF DAE]

IV.3.1.1 Cable extensions to ChannelConfig

Extension to access cable new class cableDTTChannel/cableBSCATVChannel

(1) Constant (Extension to [b-OIPF DAE], clause 7.13.10)

Constants related to this API are defined in Table IV.4.

Table IV.4 – Additional constant (broadcast network type)

Name	Value	Content
JLABS_DTT_NETWORK	0	Used in createChannelListCollection argument that indicates which network should be searched, meaning to search the digital terrestrial (DTT) network.
JLABS_BS_NETWORK	1	Used in createChannelListCollection argument that indicates which network should be searched, meaning to search the broadcast satellite (BS) network.
JLABS_CATV_NETWORK	2	Used in createChannelListCollection argument that indicates which network should be searched, meaning to search the cable TV (CATV) network.
JLABS_ALL_NETWORK	3	Used in createChannelListCollection argument that indicates which network should be searched, meaning to search *All* (including DTT/BS/CATV) networks.

(2) Method (extension to [b-OIPF DAE] clause 7.13.10)

Methods related to this API are defined in Table IV.5.

Table IV.5 – Additional method

cableChannelListCollection **createChannelListCollection** (Integer network)	
Parameters	Network
Return	cableChannelListCollection
Description	This method returns cableChannelListCollection class. The argument of "network" indicates the network which application wants to search. 0: JLABS_DTT_NETWORK 1: JLABS_BS_NETWORK 2: JLABS_CATV_NETWORK 3: JLABS_ALL_NETWORK To deal with multiple networks belonging to a same network type, this method shall be extended for JLabls.

(3) ChannelConfig element

Class elements defined by ChannelConfig are shown below. Refer to [b-OIPF DAE] for API details.

Table IV.6 – Element of ChannelConfig Class

Content	Reference to [b-OIPF DAE]
Property	
channelList	7.13.10.1
favouriteLists	7.13.10.1
currentFavouriteList	7.13.10.1
onChannelScan()	7.13.10.1
onChannelListUpdate	7.13.10.1
Method	
createFilteredList()	7.13.10.2

Table IV.6 – Element of ChannelConfig Class

Content	Reference to [b-OIPF DAE]
startScan()	7.13.10.2
stopScan()	7.13.10.2
createChannelList()	7.13.10.2
createChannelObject()	7.13.10.2
createChannelListCollection()	New
Event	
onChannelScan	7.13.10.3
onChannelListUpdate	7.13.10.3

IV.3.2 Cable extension to video/broadcast embedded object

Extend [b-OIPF DAE]standard video/broadcast embedded object to support ARIB/CATV

(1) Property (Extension to [b-OIPF DAE] clause 7.13.1.2)

Properties related to this API are defined in Table IV.7.

Table IV.7 – Additional property

function **onChannelChangeError** (Channel channel, Number errorState)	
Type	function
Description	This function is called in case of no presentation of broadcast content in the status of channel selectionerror. Following error status can be applicable to apply this property to Cable receiver. 1: cannot tune to given transport stream (e.g., no signal) ----> can display "signal lost (E202)". 5: unknown channel (e.g., cannot resolve DVB or ISDB triplet) ----> can display "non-existing channel (E204)". Considering the case that the channel is out of service, the following errorState codes shall be extended for Cable receiver: 400 : The signal level is waning although not being completely lost. : (E201) 500 : Break-1 : The service to be selected is not written in PAT : (E203) 501 : Break-2 : PMT cannot be detected although its service is written in PAT : (E203) 502 : Break-3 : The temporary service selected is not written in PAT : (E203) 503 : No signal : PAT cannot be detected although tuner is synchronized. : (E203) If these error is notified, application shall/can display an error message dialogue mentioning, for example, "the channel you selected is out of service" for E203 cases. In case of E201, the possible error message dialogue has the context "The signal level is reduced" (quoted from TR-B14) or "The video has been switched to the lower layer" (quoted from TR-B15)
function **onEmmAutoDisplayMessageNotify**(cableEmmAutoDisplayMessage ADMinfo)	
Type	function
Description	This property is for application to register the callback function in order to receive the notification of auto display message status change.

Table IV.7 – Additional property

function **onEmmNotifyNewMessage** (Integer type, Integer msgId, cableEmmMessage newMsg)	
Type	function
Description	This property is for application to register the callback function in order to know a new EMM/Internal mail being notified. – type 0: EMM mail 1: Internal mail – msgId – newMsg
function **onDRMErrorNotify** (Integer errorNumber, String DRMSystemID)	
Type	function
Description	By registering a callback function to this property, application can catch an error relating to CA. The 1st parameter "errorNumber" and the 2nd parameter "DRMSystemID" shall be defined as follows: Integer errorNumber: 2100: Normal termination (success to tune) 4480: Available for purchase: Previewing for Prepaid PPV 4280: Available for purchase: Previewing for Payment-deferred PPV 6400: Memory Scrambling Detected 6851: Memory Write Error 8500: Available for purchase: Outside preview for Prepaid PPV 8300: Available for purchase: Outside preview for Payment-deferred PPV 8901: No contract: Outside contract for Tier 8501: No contract: Outside contract for Prepaid PPV 8301: No contract: Outside contract for Payment-deferred PPV 8902: No contract: Expired for Tier 8502: No contract: Expired for Prepaid PPV 8302: No contract: Expired for Payment-deferred PPV 8903: No contract: Viewing restriction for Tier 8503: No contract: Viewing restriction for Prepaid PPV 8303: No contract: Viewing restriction for Payment-deferred PPV 8108: Purchase refused (Purchase prohibited period) 8109: Purchase refused (Viewing history memory full) 850F: Purchase refused (Insufficient prepaid balance) 9103: Center Communication Error1 9104: Center Communication Error2 A102: Non-operational Card A103: Non-contract (w/o Kw) A104: Contract Confirmation Info Tampering Error A105: EMM Message Tampering Error A106: ECM Tampering Error A107: EMM Tampering Error A1FF: Unusable Card E100: Card is not inserted E101: Card might be broken

Table IV.7 – Additional property

	E102: Improper card is inserted EC01: Invalid card EC02: When CA_system_id is not consistent <<The following 0207 and 0208 are notified in case that DRMSystemID is "urn:arib:casystemid:7".>> 0207: Cannot watch the programme due to viewing condition. (blocked by illegal usage prevention) 0208: Cannot watch the programme due to viewing condition. (illegal usage prevention : no addressable info) DRMSystemID urn:arib:casystemid:3 : conditional access system(JCL SPEC-005) urn:arib:casystemid:4 : conditional access system(JCL SPEC-005) urn:arib:casystemid:5 : conditional access system(ARIB TR-B14/B15) urn:arib:casystemid:6 : conditional access system(JCL SPEC-005) urn:arib:casystemid:7 : access control system(JCL SPEC-001-01,002,007) urn:arib:casystemid:D : Marlin IPTV-ES
function **onBmlError** (Integer errorNumber)	
Type	function
Description	This property is for registering call back function to collect errors related to BML. errorNumber (integer) 400 : The BML document cannot be obtained. 401 : The BML engine of the receiver does not support the version of the BML document obtained. 402 : During the display of the contents, an execution error has occurred and external reference data cannot be obtained.

(2) Event (extension to [b-OIPF DAE] clause 7.13.1.4)

DOM events related to this API are defined in Table IV.8.

Table IV.8 – Additional event to video/broadcast

onDRMRightsError Notify	
Description	DOM 2 Event: DRMRightsError Notify
onEmmAutoDisplayMessageNotify	
Description	DOM 2 Event: EmmAutoDisplayMessageNotify
onEmmNotifyNewMessage	
Description	DOM 2 Event: EmmNotifyNewMessage

(3) Element of video/broadcast embedded object

Element defined in video/broadcast embedded object are shown in Table IV.9. For details of [b-OIPF DAE] standard API refer to [b-OIPF DAE].

Table IV.9 – Element of video/broadcast embedded object

Content	**Reference to [b-OIPF DAE]**
Property	
width	7.13.1.2
height	7.13.1.2
fullScreen	7.13.1.2
onChannelChangeError()	7.13.1.2
playState	7.13.1.2
onPlayStateChange()	7.13.1.2
onChannelChangeSucceeded()	7.13.1.2
onFullScreenChange()	7.13.1.2
onfocus()	7.13.1.2
onblur()	7.13.1.2
programmes	7.13.3
onProgrammesChanged()	7.13.3
onParentalRatingChange()	7.13.5
onParentalRatingError()	7.13.5
onDRMRightsError()	7.13.6
onChannelChangeError()	New
onEmmAutoDisplayMessageNotify()	New
onEmmNotifyNewMessage()	New
onDRMErrorNotify()	New
onBmlError()	New
Method	
getChannelConfig()	7.13.1.3
bindToCurrentChannel()	7.13.1.3
createChannelObject ()	7.13.1.3
createChannelObject ()	7.13.1.3
setChannel ()	7.13.1.3
prevChannel ()	7.13.1.3
nextChannel()	7.13.1.3
stop()	7.13.1.3
setFullScreen()	7.13.1.3
setVolume()	7.13.1.3
getVolume()	7.13.1.3
release()	7.13.1.3
addStreamEventListener()	7.13.21

Table IV.9 – Element of video/broadcast embedded object

Content	Reference to [b-OIPF DAE]
removeStreamEventListener()	7.13.21
Event	
onfocus	7.13.1.4
onblur	7.13.1.4
onFullScreenChange	7.13.1.4
onChannelChangeError	7.13.1.4
onChannelChangeSucceeded	7.13.1.4
onPlayStateChange	7.13.1.4
onProgrammesChanged	7.13.3.1
onParentalRatingChange	7.13.5.1
onParentalRatingError	7.13.5.1
onDRMRightsError	7.13.6.1
onDRMRightsErrorNotify	New
onEmmAutoDisplayMessageNotify	New
onEmmNotifyNewMessage	New

IV.3.3 New cable extension

IV.3.3.1 cableChannelListCollection Class

This is a collection class to summarize cableDTTChannelList/cableBSCATVChannelList which is a new cable class.

(1) Constant

Constants related to this class are defined in Table IV.10.

Table IV.10 – Constant of cableChannelListCollection

Name	Value	Content
JLABS_DTT_NETWORK	0	Indicates which network (DTT/BS/CATV) belongs to cableDTTChannelList or cableBSCATVChannelList class summarized by cableChannelListCollection. Used in networkTypeName property that means this network belongs to Digital Terrestrial (DTT) Network.
JLABS_BS_NETWORK	1	Indicates which network (DTT/BS/CATV) belongs to cableDTTChannelList or cableBSCATVChannelList class summarized by cableChannelListCollection. Used in networkTypeName property that means this network belongs to Broadcast Satellite (BS) Network.
JLABS_CATV_NETWORK	2	Indicates which network (DTT/BS/CATV) belongs to cableDTTChannelList or cableBSCATVChannelList class summarized by cableChannelListCollection. Used in networkTypeName property that means this network belongs to Cable TV (CATV) Network.

(2) Property

Properties related to this class are defined in Table IV.11.

Table IV.11 – Property of cableChannelListCollection

readonly Integer **length**	
Type	readonly Integer
Description	Length of labsChannelListCollection
readonly String **networkTypeName**	
Type	readonly String
Description	Indicates BS/DTT/CATV network 0: JLABS_DTT_NETWORK：Terrestrial Digital Broadcast 1: JLABS_BS_NETWORK：BS Digital Broadcast 2: JLABS_CATV_NETWORK：CATV Digital Broadcast
readonly Integer **lastUpdateDate**	
Type	readonly Integer
Description	This property means when the latest channelList networklist can be created (scanned). UTC time format after 1970/1/1

(3) Method

Methods related to this class are defined in Table IV.12.

Table IV.12 – Method of cableChannelListCollection

ChannelList **item**(Integer num)	
Parameters	num
Return	ChannelList
Description	Return ChannelList object according to the index of arrays

IV.3.3.2 cableDTTChannelList Class

This is a collection class of cableDTTChannel class newly defined for ARIB Terrestrial Digital Broadcast.

(1) Property

Properties related to this class are defined in Table IV.13.

Table IV.13 – Property of cableDTTChannelList

readonly Integer **length**	
Type	readonly Integer
Description	Length of cableDTTChannelList
readonly String **name**	
Type	readonly String
Description	This property indicates TS name. In case of DTT, it can be used for application to display broadcaster name on UI. (NHK-G, NHK-E, NTV, TV-Asahi etc.)

Table IV.13 – Property of cableDTTChannelList

readonly Integer **areaCode**	
Type	readonly Integer
Description	This property indicates the area of DTT network based on ARIB TR-B14 Vol.7 9.1. (c) assignment of Region ID.
readonly String **signalLevel**	
Type	readonly String
Description	This property indicates the RF signal strength which was occurred in the latest network group channelist scan.
readonly Integer **branchNumber**	
Type	readonly Integer
Description	This property means the branch number if three digit channel number/oneTouch assigned to multiple DTT networks to identify the network.
Integer **oneTouch**	
Type	Integer
Description	The number that represents one touch channel.

(2) Method

Methods related to this class are defined in Table IV.14.

Table IV.14 – Method of cableDTTChannelList

cableDTTChannel **getChannel** (String channelID)	
Parameters	channelID
Return	cableDTTChannel
Description	Return the first channel in the list with the specified channel identifier. Returns null if no corresponding channel can be found. (Refer to OIPF-DAE2 7.13.11.1)
cableDTTChannel **getChannelByTriplet** (Integer onid, Integer tsid, Integer sid)	
Parameters	onid, tsid, sid
Return	cableDTTChannel
Description	Return the first (IPTV or non-IPTV) channel in the list that matches the specified DVB or ISDB triplet (original network ID, transport stream ID, service ID). Where no channels of type ID_ISDB_* or ID_DVB_* are available, or no channel identified by this triplet are found, this method SHALL return null. (Refer to OIPF-DAE2 7.13.11.1)
cableDTTChannel **item**(Integer number)	
Parameters	number
Return	cableDTTChannel
Description	Return cableDTTChannel object according to the index of arrays.

IV.3.3.3 cableDTTChannel Class

This is a newly defined channel class for terrestrial digital broadcasting.

(1) Property

Properties related to this class are defined in Table IV.15.

Table IV.15 – Property of cableDTTChannel

readonly Integer **idType**	
Type	readonly Integer
Description	The type of identification for the channel, as indicated by one of the ID_* constants defined in OIPF-DAE2 7.13.12.1. (Refer to OIPF-DAE2 7.13.12.2)
readonly String **tunerID**	
Type	readonly String
Description	Optional unique identifier of the tuner within the scope of the OITF that is able to receive the given channel. (Refer to OIPF-DAE2 7.13.12.2)
readonly Integer **onid**	
Type	readonly Integer
Description	DVB or ISDB original network ID (for channels of type ID_DVB_* and ID_ISDB_*); can be undefined if stream does not contain an SDT Actual. (Refer to OIPF-DAE2 7.13.12.2)
readonly Integer **tsid**	
Type	readonly Integer
Description	DVB or ISDB transport stream ID (for channels of type ID_DVB_* and ID_ISDB_*). (Refer to OIPF-DAE2 7.13.12.2)
readonly Integer **sid**	
Type	readonly Integer
Description	DVB or ISDB service ID (for channels of type ID_DVB_* and ID_ISDB_*). (Refer to OIPF-DAE2 7.13.12.2)
readonly Integer **serviceType**	
Type	readonly Integer
Description	This property means the Service Type in ARIB service list descriptor which represents, for example, 0x01 : Digital TV service 0x02 : Digital Audio service 0xA1 : Emergent Video service ... 0xA5 : Promotion Video service ... etc.
readonly Integer **threeDigit**	
Type	readonly Integer
Description	This property means the three digit channel number.
Boolean **invisible**	
Type	Boolean
Description	This property means if the channel is assigned to the skip channel. true means skip channel and false means not skip channel. (Refer to OIPF-DAE2 7.13.12.2)

(2) Method

Methods related to this class are defined in Table IV.16.

Table IV.16 – Method of cableDTTChannel

String **getField** (String fieldId)	
Parameters	fieldId
Return	String
Description	Get the value of the field referred to by fieldId that is contained in the BCG metadata for this channel. If the field does not exist, this method SHALL return undefined. (Refer to OIPF-DAE2 7.13.12.2)
String **getLogo** (Integer width, Integer height)	
Parameters	width, height
Return	String
Description	Get the URI for the logo image for this channel. The width and height parameters specify the desired width and height of the image; if an image of that size is not available, the URI of the logo with the closest available size not exceeding the specified dimensions SHALL be returned. If no image matches these criteria, this method SHALL return null. The URI returned SHALL be suitable for use as the SRC attribute in an HTML IMG element or as a background image. The URIs returned by this method will be derived from the values of the Logo elements that are children of the BCG ServiceInformation element describing the channel.

IV.3.3.4 cableBSCATVChannelList Class

This is a collection class of newly defined cableBSCATVChannel class for ARIB BS digital broadcasting.

(1) Property

Properties related to this class are defined in Table IV.17.

Table IV.17 – Property of cableBSCATVChannelList

readonly Integer **length**	
Type	readonly Integer
Description	Length of cableBSCATVChannelList
readonly String **signalLevel**	
Type	readonly String
Description	This property indicates the RF signal strength which was occurred in the latest network group channelist scan.

(2) Method

Methods related to this class are defined in Table IV.18.

Table IV.18 – Method of cableBSCATVChannelList

cableBSCATVChannel **getChannel** (String channelID)	
Parameters	channelID
Return	cableBSCATVChannel
Description	Return the first channel in the list with the specified channel identifier. Returns null if no corresponding channel can be found. (Refer to OIPF-DAE2 7.13.11.1)
cableBSCATVChannel **getChannelByTriplet** (Integer onid, Integer tsid, Integer sid)	
Parameters	onid, tsid, sid
Return	cableDTTChannel
Description	Return the first (IPTV or non-IPTV) channel in the list that matches the specified DVB or ISDB triplet (original network ID, transport stream ID, service ID). Where no channels of type ID_ISDB_* or ID_DVB_* are available, or no channel identified by this triplet are found, this method SHALL return null. (Refer to OIPF-DAE2 7.13.11.1)
cableBSCATVChannel **item**(Integer num)	
Parameters	num
Return	cableBSCATVChannel
Description	Return cableOtherChannel object according to the index of arrays.

IV.3.3.5 cableBSCATVChannel Class

This is a newly defined channel class for ARIB BS digital broadcasting.

(1) Property

Properties related to this class are defined in Table IV.19.

Table IV.19 – Property of cableBSCATVChannel

readonly Integer **idType**	
Type	readonly Integer
Description	The type of identification for the channel, as indicated by one of the ID_* constants defined in OIPF-DAE2 7.13.12.1. (Refer to OIPF-DAE2 7.13.12.2)
readonly String **tunerID**	
Type	readonly String
Description	Optional unique identifier of the tuner within the scope of the OITF that is able to receive the given channel. (Refer to OIPF-DAE2 7.13.12.2)
readonly Integer **onid**	
Type	readonly Integer
Description	DVB or ISDB original network ID (for channels of type ID_DVB_* and ID_ISDB_*); can be undefined if stream does not contain an SDT Actual. (Refer to OIPF-DAE2 7.13.12.2)
readonly Integer **tsid**	
Type	readonly Integer
Description	DVB or ISDB transport stream ID (for channels of type ID_DVB_* and ID_ISDB_*). (Refer to OIPF-DAE2 7.13.12.2)
readonly Integer **sid**	
Type	readonly Integer
Description	DVB or ISDB service ID (for channels of type ID_DVB_* and ID_ISDB_*). (Refer to OIPF-DAE2 7.13.12.2)

Table IV.19 – Property of cableBSCATVChannel

readonly String **name**	
Type	readonly String
Description	The name of the channel. Can be used for linking analogue channels without CNI. Typically, it will contain the call sign of the station (e.g., 'HBO'). For channels of type ID_DVB_* the service name is to be used. (Refer to OIPF-DAE2 7.13.12.2)
readonly Integer **serviceType**	
Type	readonly Integer
Description	This property means the Service Type in ARIB service list descriptor which represents, for example, 0x01 : Digital TV service 0x02 : Digital Audio service 0xA1 : Emergent Video service ... 0xA5 : Promotion Video service ... etc.
readonly Integer **threeDigit**	
Type	readonly Integer
Description	This property means the three digit channel number.
Boolean **invisible**	
Type	Boolean
Description	This property means if the channel is assigned to the skip channel. true means skip channel and false means not skip channel. (Refer to OIPF-DAE2 7.13.12.2)
Integer **oneTouch**	
Type	Integer
Description	The number that represents one touch channel.

(2) Method

Methods related to this class are defined in Table IV.20.

Table IV.20 – Method of cableBSCATVChannel

String **getField** (String fieldId)	
Parameters	fieldId
Return	String
Description	Get the value of the field referred to by fieldId that is contained in the BCG metadata for this channel. If the field does not exist, this method SHALL return undefined. (Refer to OIPF-DAE2 7.13.12.2)
String **getLogo** (Integer width, Integer height)	
Parameters	width, height
Return	String

Table IV.20 – Method of cableBSCATVChannel

Description	Get the URI for the logo image for this channel. The width and height parameters specify the desired width and height of the image; if an image of that size is not available, the URI of the logo with the closest available size not exceeding the specified dimensions SHALL be returned. If no image matches these criteria, this method SHALL return null. The URI returned SHALL be suitable for use as the SRC attribute in an HTML IMG element or as a background image. The URIs returned by this method will be derived from the values of the Logo elements that are children of the BCG ServiceInformation element describing the channel . (Refer to OIPF-DAE2 7.13.11.3)

IV.3.3.6 cableEmmMessageManager Class

This is a class to control EMM message, internal mail and board message.

(1) Method

Methods related to this class are defined in Table IV.21.

Table IV.21 – Method of cableEmmMessageManager

Integer **stopAutoDisplayMessage** (Integer id)	
Parameters	id
Return	Integer
Description	This method is for application to be able to stop displaying auto display message by calling it. For the parameter, it is necessary to specify the id value, which is notified via the callback function specified at the property "onEmmAutoDisplayMessageNotify". Application can stop displaying auto display message in case that the property "autoDisplayType" in "cableEmmAutoDisplayMessage" class object is "can be deleted". The return value is 0 if this method is successful. Otherwise, –1 is returned.
cableEmmMessageCollection **getAllEmmMessage** ()	
Parameters	None
Return	cableEmmMessageCollection
Description	This method is to get all EMM messages.
cableEmmMessageCollection **getAllSelfMessage** ()	
Parameters	None
Return	cableEmmMessageCollection
Description	This method is to get all internal messages.
cableBoardMessageCollection **getAllBoardMessage** ()	
Parameters	None
Return	cableBoardMessageCollection
Description	This method is to get all board messages.

IV.3.3.7 cableEmmAutoDisplayMessage Class

This is a class for representing AutoDisplayMessage.

(1) Property

Properties related to this class are defined in Table IV.22.

Table IV.22 – Property of cableEmmAutoDisplayMessage

Integer **id**	
Type	Integer
Description	This property is the id value to identify auto display message.
Integer **autoDisplayType**	
Type	Integer
Description	This property stands for the type value of auto display message as follows: 0 : can be deleted 1 : cannot be deleted 2 : erase message from display

IV.3.3.8 cableEmmMessage Class

This is a class representing EMM message and internal mail.

(1) Property

Properties related to this class are defined in Table IV.23.

Table IV.23 – Property of cableEmmMessage

readonly Integer **msgId**	
Type	readonly Integer
Description	This property stands for the identification value of a message. (Message ID)
readonly Integer **msgType**	
Type	readonly Integer
Description	This property stands for the type of the message. 0 : EMM message 1 : Internal message
readonly String **networkTypeName**	
Type	readonly String
Description	This property means where EMM message comes from. DTT/BS/CATV. In case of Internal Message, this property is set as "SELF".
readonly String **date**	
Type	readonly String
Description	This property is the date of the message created.
readonly String **msgBody**	
Type	readonly String
Description	In this property, the actual message body is set.
readonly Boolean **readDoneFlag**	
Type	readonly Boolean
Description	This property stands for the status of whether or not the message has already been read. FALSE : The message has not been read yet. TRUE : The message has already been read.

IV.3.3.9 cableEmmMessageCollection Class

This is a collection class of cableEmmMessage which is a cable new class.

(1) Property

Properties related to this class are defined in Table IV.24.

Table IV.24 – Property of cableEmmMessageCollection

readonly Integer **length**	
Type	readonly Integer
Description	Length of cableEmmMessageCollection

(2) Method

Methods related to this class are defined in Table IV.25.

Table IV.25 – Method of cableEmmMessageCollection

cableEmmMessage **item** (Integer num)	
Parameters	num
Return	cableEmmMessage
Description	Return cableEmmMessage object according to the index of arrays.
Integer **deleteMsg** (Integer id)	
Parameters	id
Return	Integer
Description	This method is to delete a particular EMM message by specifying msgId. If this method is successful, the return value is as follows: 0: successful –1: unsuccessful
Integer **readDone** (Integer id)	
Parameters	id
Return	Integer
Description	This method is to set the status of whether or not the message has already been read. It is necessary to specify msgId as the parameter. If this method is successful, the return value is as follows: 0: successful –1: unsuccessful

IV.3.3.10 cableBoardMessage Class

This is a class representing Board Message.

(1) Property

Properties related to this class are defined in Table IV.26.

Table IV.26 – Property of cableBoardMessage

readonly String **msgBody**	
Type	readonly String
Description	In this property, the actual message body is set.
readonly String **networkTypeName**	
Type	readonly String
Description	This property means where board message comes from. CS1/CS2.

IV.3.3.11 cableBoardMessageCollection Class

This is a collection class of cableBoardMessage which is a cable new class.

(1) Property

Properties related to this class are defined in Table IV.27.

Table IV.27 – Property of cableBoardMessageCollection

readonly Integer **length**	
Type	readonly Integer
Description	Length of cableBoardMessageCollection

(2) Property

Methods related to this class are defined in Table IV.28.

Table IV.28 – Method of cableBoardMessageCollection

cableBoardMessage **item** (Integer)	
Parameters	Integer
Return	cableBoardMessage
Description	Return cableBoardMessage object according to the index of the arrays.

IV.4 EPG

IV.4.1 Cable extension to [b-OIPF DAE]

IV.4.1.1Cable extensions to programme

Extension of [b-OIPF DAE] Standard Programme Class to meet ARIB/CATV specification

(1) Property(Extension to [b-OIPF DAE] clause 7.16.2.2）

Properties related to this class are defined in Table IV.29.

Table IV.29 – Additional property

readonly Integer **digitalCopyControl**	
Type	readonly Integer
Description	Indication of digital copy control information for programme 0 : copy free 2 : copy once or dubbing ten 3 : copy never This value is same as digital recording control date in ARIB SI digital copy control descriptor.
readonly Integer **analogCopyControl**	
Type	readonly Integer
Description	In case of Cable' STBs, it is mainly necessary for displaying an icon standing for the status of analogue copy control on a program. 0 : copy free 3 : copy never This value is same as digital recording control date in ARIB digital copy control descriptor. This value should be mapped to AGC or CGMS-A.
readonly Boolean **is3D**	
Type	readonly Boolean
Description	Indicates this programme is 3D content or not. TRUE means 3D and FALSE means not-3D.
readonly Boolean **isCharged**	
Type	readonly Boolean
Description	In case of Cable' STBs, it is mainly necessary for displaying an icon standing for whether or not a programme is Free or Charged. False : the programme is a free program. True : the programme is a Charged program.
readonly String **contractVerificationInfo**	
Type	readonly String
Description	This property indicates contract_verification_info in CA Contract Information descriptor in EIT or SDT. This property is needed to check if the programme is PPV and Recordable and check the PPV fee and subscription expiration date through DrmAgent object. Refer to ARIB STB-B25 clause 4.3 CA Interface section.

(2) Method (Extension to [b-OIPF DAE] clause 7.16.2)

Methods related to this class are defined in Table IV.30.

Table IV.30 – Additional method

AVComponentCollection **getComponents** (Integer componetType)	
Parameters	componentType
Return	AVComponentCollection
Description	It should be added to obtain information regarding each component. – Video ES – Audio ES – data ES for Subtitle – data ES for BML

(3) Element of programme

Class elements defined in the programme are shown in Table IV.31.

Table IV.31 – Element of programme class

Content	**Reference to [b-OIPF DAE]**
Constant	
ID_TVA_CRID	7.16.2.1
ID_DVB_EVENT	7.16.2.1
Property	
name	7.16.2.2
longName	7.16.2.2
description	7.16.2.2
longDescription	7.16.2.2
startTime	7.16.2.2
duration	7.16.2.2
channelID	7.16.2.2
episode	7.16.2.2
totalEpisodes	7.16.2.2
programmeID	7.16.2.2
programmeIDType	7.16.2.2
parentalRating	7.16.2.2
channel	7.16.2.3.1
blocked	7.16.2.3.1
showType	7.16.2.3.1
subtitles	7.16.2.3.1
isHD	7.16.2.3.1
audioType	7.16.2.3.1
isMultilingual	7.16.2.3.1
genre	7.16.2.3.1
hasRecording	7.16.2.3.1
audioLanguages	7.16.2.3.1
subtitleLanguages	7.16.2.3.1

Table IV.31 – Element of programme class

Content	Reference to [b-OIPF DAE]
locked	7.16.2.3.1
digitalCopyControl	New
analogCopyControl	New
is3D	New
isCharged	New
contractVerificationInfo	New
Method	
getField()	7.16.2.3.2
getComponents()	New

IV.4.1.2 Cable Extensions to AVVideoComponent

Extension of [b-OIPF DAE] Standard AVVideoComponent Class to meet ARIB/CATV specification

(1) Property(Extension to [b-OIPF DAE] clause 7.16.5.3.1)

Properties related to this class are defined in Table IV.32.

Table IV.32 – Additional property

readonly Integer **componentType**	
Type	readonly Integer
Description	This property represents Component Type in ARIB Component descriptor that means, for example, Video resolution, Aspect ratio and Audio mode. It should be added because property "aspectRatio" is not satisfied with Cable/ARIB requirement in order to display aspectRatio etc. This value originates from component_type in Component Descriptor conveyed by EIT. This value will basically be the same as ARIB's one defined at Table 6-5 stream_content and component_type in STD B-10 (part2 6.Descriptor).

(2) Element of AVVideoComponent

Class element defined in AVVideoComponent is shown in Table IV.33.

Table IV.33 – Element of AVVideoComponent Class

Content	Reference to [b-OIPF DAE]
Property	
aspectRatio	7.16.5.3.1
componentType	New

IV.4.1.3 Cable Extensions to AVAudioComponent

Extension of [b-OIPF DAE] Standard AVAudioComponent Class to meet ARIB/CATV specification

(1) Property(Extension to [b-OIPF DAE] clause 7.16.5.4.1)

Properties related to this extension are defined in Table IV.34.

Table IV.34 – Additional property

readonly Integer **componentType**	
Type	readonly Integer
Description	This property represents Component Type in ARIB Component descriptor that means, for example, Video resolution, Aspect ratio and Audio mode. It should be added because it is necessary for application to know which type of audio is there. This value originates from component_type in Audio Component Descriptor conveyed by EIT. This value will basically be the same as ARIB's one defined at Table 6-5 stream_content and component_type in STD B-10 (part2 6.Descriptor).
readonly Integer **streamType**	
Type	readonly Integer
Description	This value originates from stream_type in Audio Component Descriptor conveyed by EIT. This value will basically be the same as ARIB's one defined at Table 6-5 stream_content and component_type in STD B-10 (part2 6.Descriptor).
readonly String **language2**	
Type	readonly String
Description	An ISO 639 language code representing the language of the stream.

(2) Element of AVAudioComponent

Class elements defined in AVAudioComponent are shown in Table IV.35.

Table IV.35 – Element of AVAudioComponent Class

Content	Reference to OIPF-DAE2
Property	
language	7.16.5.4.1
audioDescription	7.16.5.4.1
audioChannels	7.16.5.4.1
componentType	New
streamType	New
language2	New
componentType	New

IV.4.1.4 Cable extensions to Media playback

Extension of AV Control Object Class defined in CEA-2014 specification to meet ARIB/CATV specification

(1) Constant（Extension to [b-OIPF DAE] clause 7.16.5.1.1)

Constants related to this extension are defined in Table IV.36.

Table IV.36 – Additional constant

Name	Value	Content
COMPONENT_TYPE_VIDEO	0	Represents a video component. This constant is used for all video components regardless of encoding.
COMPONENT_TYPE_AUDIO	1	Represents an audio component. This constant is used for all audio components regardless of encoding.
COMPONENT_TYPE_SUBTITLE	2	Represents a subtitle component. This constant is used for all subtitle components regardless of encoding.
COMPONENT_TYPE_SUPERIMPOSE	3	Represents a superimpose component. This constant is used for all superimpose components regardless of encoding.
COMPONENT_TYPE_DATA	4	Represents a data component. This constant is used for all data components regardless of encoding.

(2) Method (Extension to [b-OIPF DAE] clause 7.16.5.1.3)

Methods related to this extension are defined in Table IV.37.

Table IV.37 – Additional method

AVComponentCollection **getComponents** (integer componetType)	
Parameters	componentType
Return	AVComponentCollection
Description	Returns a collection of AVComponent values representing the components of the specified type in the current stream. If componentType is set to null or undefined then all the currently active components are returned.

(3) Element of MediaPlaybackExtentions

Class elements defined in MediaPlayback are shown in Table IV.38.

Table IV.38 – Element of MediaPlaybackExtentions

Content	Reference to OIPF-DAE2
Constant	
COMPONENT_TYPE_VIDEO	7.16.5.1.1
COMPONENT_TYPE_AUDIO	7.16.5.1.1
COMPONENT_TYPE_SUBTITLE	7.16.5.1.1
COMPONENT_TYPE_VIDEO	New
COMPONENT_TYPE_AUDIO	New
COMPONENT_TYPE_SUBTITLE	New
COMPONENT_TYPE_SUPERIMPOSE	New
COMPONENT_TYPE_DATA	New
Property	
onSelectedComponentChanged()	7.16.5.1.2
Method	

Table IV.38 – Element of MediaPlaybackExtentions

Content	Reference to OIPF-DAE2
getComponents()	7.16.5.1.3
getCurrentActiveComponents ()	7.16.5.1.3
selectComponent ()	7.16.5.1.3
unselectComponent ()	7.16.5.1.3
unselectComponent ()	7.16.5.1.3
selectComponent ()	7.16.5.1.3
unselectComponent()	7.16.5.1.3
getComponents()	New
Event	
onSelectedComponentChange	7.16.5.1.4

IV.4.2 New CableExtension

IV.4.2.1 cableDataContentComponent Class

Defines new data component class to meet BML specification

(1) Property

Properties related to this class are defined in Table IV.39.

Table IV.39 – Property of cableDataContentComponent

readonly Boolean **associatedContentFlag**	
Type	readonly Boolean
Description	This property means whether or not the data ES containing BML associates with TV program. – True : The data ES (containing BML) which associates with TV programme is broadcasting. – False : The data ES (containing BML) which does not associate with TV programme is broadcasting. This value originates from associated_content_flag in Data Content Descriptor conveyed by EIT.
readonly Integer **digitalCopyControl**	
Type	readonly Integer
Description	This value is set in case that the different copy control policy from Video/Audio/Data containing subtitle/superimpose is applied for Data ESs containing BML. 0 : copy free 2 : copy once or dubbing ten 3 : copy never This digital copy control property will override the original value defined in the programme class object.

IV.5 Set Up

IV.5.1 New Cable Extension

IV.5.1.1cableNetworkInterface Class

Defines [b-OIPF DAE]Standard NetworkInterface class newly to meet Ipv6 specification

(1) Property

Properties related to this class are defined in Table IV.40.

Table IV.40 – Property of cableNetworkInterface

readonly Boolean **ipv6Type**	
Type	readonly Boolean
Description	Readonly Flag to denote whether the Network Interface client is using IPv6 or not. TRUE means IPv6 and FALSE means IPv4 network.
String **interfaceType**	
Type	String
Description	Name of network interface
Boolean **enable**	
Type	Boolean
Description	flag to enable/disable a network interface
Boolean **dhcpEnable**	
Type	Boolean
Description	Flag to denote whether DHCP is enabled or not. TRUE means to use DHCP address discovery and FALSE means to use static setting.
readonly String **macAddress**	
Type	readonly String
Description	MAC Address
String **ipv4Address**	
Type	String
Description	IP Address of network interface
String **subnetMask**	
Type	String
Description	Sub net mask of network interface
String **defaultGateway**	
Type	String
Description	Denotes the default gateway address for the network interface. The address length will differ according to ipv6Type.
String **primaryDNSServerAddress**	
Type	String
Description	This property means the Primary DNS Server IP address. The address length will differ according to iv6Type.

Table IV.40 – Property of cableNetworkInterface

String **secondaryDNSServerAddress**	
Type	String
Description	Denotes the secondary DNS server IP address. The address length will differ according to ipv6Type.
function **onlinkStatusChanged**	
Type	function
Description	notification "LinkStatusChanged" event for status for link up/down
String **ipv6Address**	
Type	String
Description	IPv6 IP Address if interface
String **tunnelStartIpv4Address**	
Type	String
Description	IPv6 specific property, specifies the starting tunnel address in IPv4 format.
String **tunnelEndIpv4Address**	
Type	String
Description	IPv6 specific property, specifies the ending tunnel address in IPv4 format.
Integer **prefixLength**	
Type	Integer
Description	IPv6 specific property, specifies the prefix length for IPv6 network interface.

(2) Method

Methods related to this class are defined in Table IV.41.

Table IV.41 – Method of cableNetworkInterface

String **get** (String name)	
Parameters	name
Return	String
Description	get the property
String **set** (String name, String value)	
Parameters	name, value
Return	String
Description	set the property
Integer **restat**( )	
Parameters	None
Return	Integer
Description	restart the network interface with new settings. If SUCCESS then new settings shall apply and if FAILURE, the previous setting will be restored. The return value shall be 0 if SUCCESS and below zero if FAILURE.

Table IV.41 – Method of cableNetworkInterface

Boolean **isDown**( )	
Parameters	None
Return	Boolean
Description	check if the network interface is down. TRUE means network interface down and FALSE means other than down.
Boolean **isUp**( )	
Parameters	None
Return	Boolean
Description	check if the network interface is up. TRUE means the network interface is up and FALSE means other than that.

(2) Event

Event related to this class is defined in Table IV.42.

Table IV.42 – Event of cableNetworkInterface

LinkStatusChanged	
Description	LinkStatusChanged event is notified with the status

IV.5.1.2 cableNetworkInterfaceCollection Class

Collection Class of cableNetworkInterface which is new Cable class

(1) Property

Properties related to this class are defined in Table IV.43.

Table IV.43 – Property of cableNetworkInterfaceCollection

Integer **length**	
Type	Integer
Description	Length of cableNetworkInterfaceCollection

(2) Method

Methods related to this class are defined in Table IV.44.

Table IV.44 – Method of cableNetworkInterfaceCollection

Object **item** (Integer number)	
Parameters	Integer
Return	Object
Description	Return cableNetworkInterface object in indicated position

IV.5.1.3 cableNetworkService Class

This is a class to be extended for wireless LAN, cable modem, WAN and WiFi, DHCP server, NAT, or firewall setting. Each entity property should not be defined for the future extension. Only offer the serviceName and targetName to set the property values. For example, for DHCP server service, serviceName should be set to "DHCPServer" and targetName should be set to "192.168.1.1, 24". This means DHCP client addresses should be delivered from 192.168.1.1 to 192.168.1.24.

(1) Property

Properties related to this class are defined in Table IV.45.

Table IV.45 – Property of cableNetworkService

readonly String **serviceName**	
Type	readonly String
Description	name of the service. like "DHCP", "Router", "WiFiAccessPoint"."NAT" or "Firewall".
String **targetName**	
Type	String
Description	name of the server interface. like NetworkInterface.name

(2) Method

Methods related to this class are defined in Table IV.46.

Table IV.46 – Method of cableNetworkService

String **get** (String name)	
Parameters	name
Return	String
Description	Get the property
Integer **set** (String name, String value)	
Parameters	name, value
Return	Integer
Description	Set the property
Integer **restart** ()	
Parameters	None
Return	Integer
Description	restart the network services with new settings. If SUCCESS then new settings shall apply.

IV.5.1.4 cableNetworkServiceCollection Class

Collection class of cableNetworkService which is cable new class

(1) Property

Properties related to this class are defined in Table IV.47.

Table IV.47 – Property of cableNetworkServiceCollection

Integer **serviceName**	
Type	Integer
Description	Length of cableNetworkServiceCollection

(2) Method

Methods related to this class are defined in Table IV.48.

Table IV.48 – Method of cableNetworkServiceCollection

Object **get** (Integer number)	
Parameters	Integer
Return	Object
Description	Return cableNetworkService object in indicated position

IV.5.1.5 cableMiscSetting Class

Defines functions except for network, scan, PPV and parental control within cable extended setup class

(1) Constant

Constants related to this class are defined in Table IV.49.

Table IV.49 – Constant of cableMiscSetting

Name	Value	Content
JLABS_DTT_NETWORK	0	Used in networkTypeName property that means this network belongs to Digital Terrestrial(DTT) Network.
JLABS_BS_NETWORK	1	Used in networkTypeName property that means this network belongs to Broadcast Satellite(BS) Network.
JLABS_CATV_NETWORK	2	Used in networkTypeName property that means this network belongs to Cable TV(CATV) Network.

(2) Property

Properties related to this class are defined in Table IV.50.

Table IV.50 – Property of cableMiscSetting

String **postCode**	
Type	String
Description	Denotes the postal pin of the area. string length is 10, but 7 numerals shall be sanity checked by application
String **localArea**	
Type	String
Description	Denotes the local area like Tokyo etc.
Boolean **autoSoftwareDownloadEnable**	
Type	Boolean
Description	Denotes the Auto Software download settings. If TRUE then the software shall be downloaded automatically without any user's confirmation.
Boolean **channelMaskEnable**	
Type	Boolean
Description	Denotes the auto channel mask, if TRUE the channel's masked shall not be shown in EPG etc.

Table IV.50 – Property of cableMiscSetting

Integer **subtitleMod**e	
Type	Integer
Description	0 – NONE 1 – LANGUAGE_1 2 – LANGUAGE_2 If set to 1 or 2, and that setting is not available in stream, then the other available language shall be shown.
Integer **superimposeMode**	
Type	Integer
Description	0 – NONE 1 – LANGUAGE_1 2 – LANGUAGE_2 If set to 1 or 2, and that setting is not available in stream, then the other available language shall be shown.
String **tvAspectRatio**	
Type	String
Description	Default Value = "16:9", Aspect Ratio used for Composite Outputs
String **spdifMode**	
Type	String
Description	SPDIF settings, "AUTO"/"PCM"
readonly Integer **lastChannelNetworkTypeName**	
Type	readonly Integer
Description	This property means the last selected Network type, like DTT/BS/CATV network. 0: JLABS_DTT_NETWORK 1: JLABS_BS_NETWORK 2: JLABS_CATV_NETWORK
readonly Integer **lastChannelThreeDigitNumber**	
Type	readonly Integer
Description	This property means the last selected channel three digit numbers, same value as the property of "threeDigit" in cableDTTChannel or cableBSCATVChannel.
Boolean **seamlessEnable**	
Type	Boolean
Description	Seamless mode definition: If TRUE, three digit input, channel up/down tuning are performed among the three network service(DTT/BS/CableTV). If FAULSE, three digit input, channel up/down tuning are performed within the current network service. On the other hand, one touch tuning is always performed within the current network service.

Table IV.50 – Property of cableMiscSetting

Integer **setupConfigurationLaunchMode**	
Type	Integer
Description	can be 0 – NORMAL 1 – WIZARD 2 – INSTALLER
Boolean autoPPVPurchaseEnable	
Type	Boolean
Description	Flag to enable/disable PPV auto purchase
Integer **3DdisplayMode**	
Type	Integer
Description	This property controls how 3D broadcast shall be overlapped with 2D graphic according to JCL-SPEC-023.
Integer **numberOfTuners**	
Type	Integer
Description	This property specifies how many tuner resources are available in the device. It is for the application to configure this setting to let the underlying conflict manager decide on conflict priorities accordingly.

(3) Method

Methods related to this class are defined in Table IV.51.

Table IV.51 – Method of cableMiscSetting

String **get** (String name)	
Parameters	name
Return	String
Description	Method to get property
Integer **set** (String name, String value)	
Parameters	name, value
Return	Integer
Description	Method to set property

IV.5.1.6 cableSystemInfo Class

Setup class to define STB fixed value

(1) Property

Properties related to this class are defined in Table IV.52.

Table IV.52 – property of cableSystemInfo

readonly String **cardIdBCAS**	
Type	readonly String
Description	B-CAS Card ID
readonly String **cardIdCCAS**	
Type	readonly String
Description	C-CAS Card ID
readonly Boolean **cardStatusBCAS**	
Type	readonly Boolean
Description	Status of B-CAS card
readonly Boolean **cardStatusCCAS**	
Type	readonly Boolean
Description	Status of C-CAS card
readonly String **deviceId**	
Type	readonly String
Description	Device ID
readonly String **deviceModel**	
Type	readonly String
Description	Device Model name
readonly String **deviceSerialNumber**	
Type	readonly String
Description	Device serial number
readonly String **deviceSoftwareVersion**	
Type	readonly String
Description	Current software version
readonly String **deviceHardwareVersion**	
Type	readonly String
Description	Current hardware version
readonly Integer **systemUpdateDate**	
Type	readonly Integer
Description	Readonly System's Last Update Date. UTC time format after 1970/1/1.

(2) Method

Methods related to this class are defined in Table IV.53.

Table IV.53 – Method of cableSystemInfo

String **get** (String name)	
Parameters	name
Return	String
Description	Method to get property

IV.5.1.7 cablePPVHistory Class

Class to get PPVhistory

(1) Property

Properties related to this class are defined in Table IV.54.

Table IV.54 – Property of cablePPVHistory

Integer **date**	
Type	Integer
Description	Date of the PPV Purchase made. UTC time format after 1970/1/1.
String **fee**	
Type	String
Description	Fee of the PPV Purchase made.
String **eventData**	
Type	String
Description	Metadata of the event purchased.
readonly String **caSystemId**	
Type	readonly String
Description	CAS System ID. This shall be mapped to DRM system ID.

(2) Method

Methods related to this class are defined in Table IV.55.

Table IV.55 – Method of cablePPVHistory

String **get** (String name)	
Parameters	name
Return	String
Description	Method to get property
Integer **set** (String name, String, value)	
Parameters	name, value
Return	Integer
Description	Method to set property

IV.5.1.8 cablePPVHistoryCollection Class

Collection class of cablePPVHistory which is new Cable class

(1) Property

Properties related to this class are defined in Table IV.56.

Table IV.56 – Property of cablePPVHistoryCollection

readonly Integer **totalPurchasedFee**	
Type	readonly Integer
Description	Readonly denotes total PPV purchase done so far after last clearance
readonly Integer **length**	
Type	readonly Integer
Description	Length of cablePPVHistoryCollection

(2) Method

Methods related to this class are defined in Table IV.57.

Table IV.57 – Method of cablePPVHistoryCollection

Integer **uploadPPVViewlogData** ()	
Parameters	None
Return	Integer
Description	Method to upload the PPV Viewlog Data. Returns SUCCESS(0)/FAILURE(-1)
Integer **deleteAllPPVPrivateData** ()	
Parameters	None
Return	Integer
Description	Method to delete the private accumulated PPV History Data from the STB.
PPVHistoryCollection **getPPVPurchaseHistory** ()	
Parameters	None
Return	PPVHistoryCollection
Description	Method to get the PPV Purchase History data until last clearance.

IV.5.1.9 cableLocalSystem Class

New class to collect cableNetworkInterfaceCollection, cableNetworkServiceCollection, cableMiscSetting, cablePPVHistoryCollection, cableScanningCollection, cableSystemInfo

(1) Property

Properties related to this class are defined in Table IV.58.

Table IV.58 – Property of cableLocalSystem

cableNetworkInterfaceCollection **networkInterfaces**	
Type	cableNetworkInterfaceCollection
Description	Element of cableNetworkInterfaceCollection
cableNetworkServiceCollection **networkService**	
Type	cableNetworkServiceCollection
Description	This class is the collection of NetworkService class and NetworkService class, which is intended to create for Cable Extension, will defines various network service entities. For example, DHCP server service, WiFi server service, NAT/firewall service, etc.
cableMiscSetting **misc**	
Type	cableMiscSetting

Table IV.58 – Property of cableLocalSystem

Description	Element of cableMiscSetting
readonly cableSystemInfo **systemInfo**	
Type	readonly cableSystemInfo
Description	Element of cableSystemInfo
PPVHistoryCollection **ppvHistory**	
Type	PPVHistoryCollection
Description	Element of PPVHistoryCollection
cableScanningCollection **scanComponent**	
Type	cableScanningCollection
Description	Element of cableScanningCollection

(2) Method

Methods related to this class are defined in Table IV.59.

Table IV.59 – Method of cableLocalSystem

Integer **openBrowser**(String url)	
Parameters	url
Return	Integer
Description	Open URL indicated to make browser test, return value are as follows; 0: SUCCESS –1: FAILURE
Boolean **ping**(String addr)	
Parameters	addr
Return	Boolean
Description	Ping to indicated Address, return value are as follows; 0: SUCCESS –1: FAILURE
Integer **setFactoryReset** ()	
Parameters	None
Return	Integer
Description	Recover to default setting (Factory reset), return value are as follows; 0: SUCCESS –1: FAILURE

IV.5.1.10 cableScanning Class

Setting class for channel scanning

(1) Property

Properties related to this class are defined in Table IV.60.

Table IV.60 – Property of cableScanning

readonly String **networkType**	
Type	readonly String
Description	Network identification (DTT/BS/CATV)
ChannelScanParameter **channelScanParameter**	
Type	ChannelScanParameter
Description	Element of ChannelScanParameter
ChannelScanOptions **channelScanOptions**	
Type	ChannelScanOptions
Description	Element of ChannelScanOption
String **favId**	
Type	String
Description	This property indicates the Favourite ID in the Favourite list class. If the value is NULL then no favourite list exists.

IV.5.1.11 cableScanningCollectionClass

Collection class of cableScanning which is cable new class

(1) Property

Properties related to this class are defined in Table IV.61.

Table IV.61 – Property of cableScanningCollection

Integer **length**	
Type	Integer
Description	Length of jlabScanningcollection

(2) Method

Methods related to this class are defined in Table IV.62.

Table IV.62 – Method of cableScanningCollection

Object **item**(Integer number)	
Parameters	number
Return	Object
Description	Get cableScanning object in indicated position

IV.6 Recording/home network

IV.6.1 Extension to [b-OIPF DAE]

IV.6.1.1 Cable Extension to RecordingScheduler

Extension of [b-OIPF DAE] standard RecordingScheduler class to meet ARIB/CATV specification

(1) Constant (Extension to [b-OIPF DAE] clause 7.10.1)

Constants related to this extension are shown in Table IV.63.

Table IV.63 – Additional constant to RecordingScheduler

Name	Value	Content
USB_RECORDING	0	Indicated reservation is recording to USB storage device
VIEWING_RESERVATION	1	Indicated reservation is watching reservation
DLNA_UPLOAD	2	Indicated reservation is upload to DLNA DMS device

(2) Property (Extension to [b-OIPF DAE] clause 7.10.1)

Properties related to this extension are shown in Table IV.64.

Table IV.64 – Additional Property to RecordingScheduler

function **onHNReservationFinished**(ScheculedRecording recording, Integer errorCode)	
Type	function
Description	This function is the DOM 0 event handler for notification of the result for DLNA reservation. DOM 2 event name is HNReservationFinished. scheduledRecording: DLNA ScheduledRecording object which was created at record() or recordAt() method. errorCode: The reservation result 0 : success non 0: failure This event will be notified after record() and recordAt() methods are called for the result of DLNA reservation. For DLNA reservation, both the local scheduler booking and the DMS scheduler booking are required and the DMS reservation result will come back as a SOAP response.

(3) Method (Extension to [b-OIPF DAE] clause 7.16.10.1)

Methods related to this extension are shown in Table IV.65.

Table IV.65 – Additional method

ScheduledRecording **record** (Programme programme, Integer factory)	
Parameters	programme, factory
Return	ScheduledRecording
Description	Requests the scheduler to schedule the recording of the programme identified by the programmeID property of the programme. The other data contained in the programme Object is used solely for annotation of the (scheduled) recording. If such programme metadata is provided, it is retained in the ScheduledRecording object that is returned if the recording of the programme was scheduled successfully, reflecting the possibility that not all relevant metadata might be available to the scheduler. If the recording could not be scheduled due to a scheduling conflict or lack of resources the value null is returned. Factory is specified whether recording/viewing reservation or DLNA reservation. If the factory argument is DLNA_UPLOAD, reservation needs both acknowledgements from local and remote (DMS) scheduler and the response will delay, so application needs to wait for the event of onHNReservationFinished and confirm the result.
ScheduledRecording **recordAt** (Integer startTime, Integer duration, Integer repeatDays, String ChannelID, Integer factory)	
Parameters	startTime, duration, repeatDays, ChannelID, factory
Return	ScheduledRecording

Table IV.65 – Additional method

Description	Requests the scheduler to schedule the recording of the broadcast to be received over the channel identified by channelID, starting at startTime and continuing for duration minutes. If the recording was scheduled successfully, the resulting ScheduledRecording object is returned. If the recording could not be scheduled due to a scheduling conflict or lack of resources the value null is returned. Factory is specified whether recording/viewing reservation or DLNA reservation. If the factory argument is DLNA_UPLOAD, reservation needs both acknowledgements from local and remote(DMS) scheduler and the response will delay, so application needs to wait for the event of onHNReservationFinished and confirm the result.
RecordingCollection **getRecordings** ()	
Parameters	None
Return	RecordingCollection
Description	Get RecordingCollectionobject
Integer **setDefaultDMS** (String deviceHandle)	
Parameters	deviceHandle
Return	Integer
Description	This method is used to set the default cableDMS object to be uploaded for the content. deviceHandle: The property of "deviceHandle" in cableCDS object to be set as the default DMS Return value: Integer 0 : success non 0 : failure

(4) Event (Extension to [b-OIPF DAE] clause 7.10.1)

The event related to this extension is shown in Table IV.66.

Table IV.66 – Additional event

onHNReservationFinished	
Description	Equivalent DOM2 event: HNReservationFinished

(5) Element of application/oipfRecordingScheduler embedded object

Elements defined in application/oipfRecordingScheduler embedded object are shown in Table IV.67.

Table IV.67 – Element of application/oipfRecordingScheduler embedded object

Content	Reference to OIPF-DAE2
Property	
recordings	7.10.4.1
discInfo	7.10.4.1
onPVREvent()	7.10.4.1
onHNReservationFinished()	New
Method	
getScheduledRecordings()	7.10.1.1
getChannelConfig()	7.10.1.1

Table IV.67 – Element of application/oipfRecordingScheduler embedded object

Content	Reference to OIPF-DAE2
remove()	7.10.1.1
createProgrammeObject()	7.10.1.1
getRecording()	7.10.4.2
stop()	7.10.4.2
refresh()	7.10.4.2
record()	New
recordAt()	New
getRecordings()	New
setDefaultDMS()	New
Event	
onPVREvent	7.10.4.3
onHNReservationFinished	New

IV.6.1.2 Cable Extension to Recording

Extension of [b-OIPF DAE] Standard Recording to meet ARIB/CATV specification.

(1) Property (Extension to [b-OIPF DAE] clause 7.10.5)

Properties defined in Recording Class are shown in Table IV.68.

Table IV.68 – Additional property

readonly Integer **digitalCopyControl**	
Type	readonly Integer
Description	Property signifying digital copy protection information on the recording. 0 : copy free 2 : copy once or dubbing ten 3 : copy never
readonly Integer **analogCopyControl**	
Type	readonly Integer
Description	Property signifying analogue copy protection information on the recording. 0 : copy free 3 : copy never

(2) Element of recording class

Elements defined in recording class are shown in Table IV.69.

Table IV.69 – Element of recording class

Content	Reference to OIPF-DAE2
Property	
state	7.10.5.1
id	7.10.5.1
isManual	7.10.5.1
doNotDelete	7.10.5.1

Table IV.69 – Element of recording class

Content	Reference to OIPF-DAE2
saveDays	7.10.5.1
saveEpisodes	7.10.5.1
showType	7.10.5.1
subtitles	7.10.5.1
subtitleLanguages	7.10.5.1
isHD	7.10.5.1
isWidescreen	7.10.5.1
audioType	7.10.5.1
isMultilingual	7.10.5.1
audioLanguages	7.10.5.1
genres	7.10.5.1
recordingStartTime	7.10.5.1
recordingDuration	7.10.5.1
bookmarks	7.10.5.1
locked	7.10.5.1
digitalCopyControl	New
analogCopyControl	New

IV.6.1.3 Cable Extension to ScheduledRecording

Extension of [b-OIPF DAE] ScheduledRecording Class to meet DLNA/watching reservation requirement

(1) Property (Extension to [b-OIPF DAE] clause 7.10.5)

Properties defined in ScheduledRecording Class are shown in Table IV.70.

Table IV.70 – Additional Property

Integer **factoryType**	
Type	Integer
Description	Property signifying whether the recording or viewing reservation. USB_RECORDING: Recording to internal HDD VIEWING_RESERVATION: Viewing Reservation DLNA_UPLOAD: Upload to the default DMS

(2) Element of ScheduledRecording Class

Elements defined in ScheduledRecording Class are shown in Table IV.71.

Table IV.71 – Element of ScheduledRecording Class

Content	Reference to OIPF-DAE2
Constant	
ID_TVA_CRID	7.10.2.1
ID_DVB_EVENT	7.10.2.1
ID_TVA_GROUP_CRID	7.10.2.1
Property	
startPadding	7.10.2.2
endPadding	7.10.2.2
repeatDays	7.10.2.2
name	7.10.2.2
longName	7.10.2.2
description	7.10.2.2
longDescription	7.10.2.2
startTime	7.10.2.2
duration	7.10.2.2
channel	7.10.2.2
isSeries	7.10.2.2
programmeID	7.10.2.2
programmeIDType	7.10.2.2
episode	7.10.2.2
totalEpisodes	7.10.2.2
parentalRatings	7.10.2.2
factoryType	New

IV.6.2 New Cable Extension

IV.6.2.1 cableCDSSearcher Class

Class to search DLNA DMS in home network

(1) Property

Properties related to this class are defined in Table IV.72.

Table IV.72 – Property of cableCDSSearcher

function **onDMSAttached**(cableDMSCollection dms)	
Type	function
Description	This function property is used for the event notification of cableDMS objects that are newly attached into the local network. dms: cableDMSCollection object which joined into the network This event notifies multiple cableDMS objects. Middleware is required to monitor that the multiple DMS are joined in the network with an adequate time period to avoid multiple events.

function **onDMSDetached** (cableDMSCollection dms)	
Type	function
Description	This function property is used for the event notification of cableDMS objects that are detached from the local network. dms: cableDMSCollection object which leave from the network This event notifies multiple cableDMS objects. Middleware is required to monitor that the multiple DMS are left from the network with an adequate time period to avoid multiple events.

(2) Method

Methods related to this class are defined in Table IV.73.

Table IV.73 – Method of cableCDSSearcher

cableDMSCollection **CDSSearch** (Integer timeout)	
Parameters	timeout
Return	cableDMSCollection
Description	This method is used to search the multiple cableDMS objects which are reside in the local network. timeout: second to wait for DMS SSDP:alive response Return cableDMSCollection object This is a blocking API to search cableDMS objects inside the local network.

(3) Event

Events related to this class are defined in Table IV.74.

Table IV.74 – Additional Event to cableCDSSearcher

onDMSAttached	
Description	Equivalent DOM2 event: DMSAtttached
onDMSDetached	
Description	Equivalent DOM2 event: DMSDetached

IV.6.2.2 cableDMSCollection Class

Collection class of cableDMS which is newCable class

(1) Property

Properties related to this class are defined in Table IV.75.

Table IV.75 – Property of cableDMSCollection

readonly Integer **length**	
Type	readonly Integer
Description	Length of cableDMSCollection（Number of cableDMS object）

(2) Method

Methods related to this class are defined in Table IV.76.

Table IV.76 – Method of cableDMSCollection

cableDMS **item** (Integer num)	
Parameters	num
Return	cableDMS
Description	This method returns cableDMS object. num: Integer which means the index of cableDMS objects

IV.6.2.3 cableDMSb Class

Class representing DLNA DMS

(1) Property

Properties related to this class are defined in Table IV.77.

Table IV.77 – Property of cableDMS

readonly Integer **deviceHandle**	
Type	readonly Integer
Description	This property indicates the cableDMS Object Id which identifies DMS.
readonly String **UDN**	
Type	readonly String
Description	This property indicates DMS "uuid" that is unique inside DLNA network.
String **friendlyName**	
Type	String
Description	This property indicates User friendly DMS name that can be modified by User. For example, "Father's DMS in the 2nd floor".
readonly String **ipAddress**	
Type	readonly String
Description	This property indicates DMS IP address for User to identify the DMS.
readonly String **MACAddress**	
Type	readonly String
Description	This property indicates MAC Address assigned to the DMS.
readonly String **totalStorageSize**	
Type	readonly String
Description	This property indicates the total storage size for DMS to upload. The String format is the number string unit by Byte.
readonly StringCollection **sortCAP**	
Type	readonly StringCollection
Description	This property contains various String that indicates the DMS capability to sort by Server. For example, "+date" means to sort by "date" in ascending order. This property may contain various capabilities. For example, "+date" and "+artist" means sort can be done by the metadata of "date" and "artist" in ascending order.
readonly StringCollection **searchCAP**	
Type	readonly StringCollection

Table IV.77 – Property of cableDMS

Description	This property contains various String that indicates the DMS capability to search by Server. For example, "date" means to search by "date". This property may contain various capabilities. For example, "date" and "artist" means search can be done by the metadata of "date" and "artist"
readonly StringCollection **dlnaCAP**	
Type	readonly StringCollection
Description	This property contains various String that indicates the DMS capability defined by DLNA. Refer to DLNA 1.5 guideline for the details.
readonly StringCollection **cableCAP**	
Type	readonly StringCollection
Description	This property contains various String that indicates the DMS capability defined by JLABs. Refer to JCL_SPEC-020 for the details. From Identifier A to Identifier F are defined by JLABs.
cableContentSearch **contentSearch**	
Type	cableContentSearch
Description	This property indicates cableContentSearch object to be used for Searching cableCDSRecording object if the property of "searchCAP" contains active search capability.
function **onContentSearch**	
Type	function
Description	This function property indicates the event notification when searching cableCDSRecording object is finished.

(2) Method

Methods related to this class are defined in Table IV.78.

Table IV.78 – Method of cableDMS

cableCDSRecordingCollection **browseDirectChildren** (String path, String sortOrder)	
Parameters	path, sortOrder
Return	cableCDSRecordingCollection
Description	This method is used for browsing the contents under the designated path string. path: String property that means the designate path to be browsed under this directory. sortOrder: String property that means the sort order if DMS has the capability of sort. The root path always starts from "0". The path string has hierarchy as concatenating the objectID in cableCDSRecording object to the tail of parent path. SortOrder has a presentation by the DMS metadata. For example, if DMS can sort by ascending order of "title", this is represented as "+date".
cableContentSearch **createContentSearch** (Integer timeout)	
Parameters	timeout
Return	cableContentSearch

Table IV.78 – Method of cableDMS

Description	This method is used to create cableContentSearch object to search cableCDSRecording object. This method is used when the property of "searchCAP" contains available search capability. timeout: Integer type property that means the timeout seconds to search cableCDSRecording object. The default value is set to adequate time by middleware.
readonly String **getAvailableStorageSize**()	
Parameters	None
Return	readonly String
Description	This method is used for retrieving the current DMS available storage size to upload. This is for displaying purpose, but if Application wants to calculate the proposition of the available storage size against total storage size, String to Integer conversion will be needed.

(3) Event

Event related to this class is defined in Table IV.79.

Table IV.79 – cableDMS Event

onContentSearch	
Description	ContentSearch

IV.6.2.4 cableContentSearch Class

Class for searching recorded content over DMS

(1) Property

Properties related to this class are defined in Table IV.80.

Table IV.80 – Property of cableContentSearch

Query **query**	
Type	Query
Description	This property is same as the property of [b-OIPF DAE] section 7.12.3 Query class. If DMS supports server side search capability, this query object will be used.
cableContentResults **result**	
Type	cableContentResults
Description	This property is used to get the search results.

(2) Method

Methods related to this class are defined in Table IV.81.

Table IV.81 – Method of cableContentSearch

void **setQuery** (Query query)	
Parameters	query
Return	void
Description	This method is used to set Query object to be used for Search.
void **orderBy** (String field, Boolean ascending)	

Table IV.81 – Method of cableContentSearch

Parameters	field, ascending
Return	void
Description	This method is used to set Search sort order if DMS has capability of sort. field: String type property indicating metadata property name, for example, "title". ascending: Boolean type property indicating the sort ascending order if this value is true.
Query **createQuery**(String field, Integer comparison, String value)	
Parameters	field, comparison, value
Return	Query
Description	This method creates new Query object. field: String type argument indicating the metadata property name, for example, "title". comparison: Integer type argument indicating the type of comparison. 0: True if the specified value is equal to the value of the specified field. 1: True if the specified value is not equal to the value of the specified field. 2: True if the value of the specified field is greater than the specified value. 3: True if the value of the specified field is greater than or equal to the specified value. 4: True if the value of the specified field is less than the specified value. 5: True if the value of the specified field is less than or equal to the specified value. 6: True if the string value of the specified field contains the specified value. This operation SHALL be case insensitive, and SHALL match parts of a word as well as whole words (e.g., a value of "term" will match a field value of "Terminator"). 7: True if the specified field exists value: String type argument indicating the value to be checked.

IV.6.2.5 cableContentResults Class

Class to obtain searching result of recorded content over DMS

(1) Property

Properties related to this class are defined in Table IV.82.

Table IV.82 – Property of cableContentResults

readonly Integer **length**	
Type	Readonly Integer
Description	The number of items in the currently available results. If results are fetched asynchronously, the value of this property SHALL be zero until after update() has been called.
readonly Integer **offset**	
Type	readonly Integer
Description	The current offset into the total result set.
readonly Integer **totalSize**	
Type	readonly Integer
Description	The total number of items in the result set. If results are fetched asynchronously, the value of this property SHALL be undefined until getResults() has been called and a MetadataSearchEvent notifying the application that results are available has been dispatched.

(2) Method

Methods related to this class are defined in Table IV.83.

Table IV.83 – Method of cableContentResults

cableCDSRecording **item** (Integer num)	
Parameters	num
Return	cableCDSRecording
Description	Return the item at position index with num in the collection of currently available results.
Boolean **getResults** (Integer offset, Integer count)	
Parameters	offset, count
Return	Boolean
Description	Perform the search and retrieve a subset of the items that match the query. Results MAY be returned both synchronously and asynchronously, depending on whether data is available from the cache. If getResults() returns false, results are not available until the notification events have been returned and update() has been called. If getResults() returns true, results are available immediately, and the application need not wait for MetadataSearchEvents indicating that results are available or call update() to obtain the results. For results returned as a result of the same call to getResults(), the full result set may build up over time – the availability of new entries in the result set will be indicated by notification events. Subsequent calls to getResults() will clear the result set, so only results fetched for the most recent call to getResults() will be available to applications.
void **abort**()	
Parameters	None
Return	void
Description	Abort any outstanding request for results. Items currently in the collection SHALL be removed (i.e., the value of the length property SHALL be 0 and any calls to item()SHALL return undefined).
void **update**()	
Parameters	None
Return	void
Description	Through the update method new results are made available to applications. When a call to getResults() has returned false and results are fetched asynchronously, this method must be called after an application has received a notification event informing it that new results are available. The results may be delivered over multiple notification events. Until this method is called, results returned by asynchronous requests SHALL NOT be available to applications. This ensures that applications have a consistent view of the available results, without the result set changing asynchronously. This enables applications to (for example) iterate over the current result set and update their UI before retrieving the new results which have been returned to the OITF but are not yet available to applications.

IV.6.2.6 cableCDSRecordingCollection Class

Collection class of cableCDSRecording which is Cable new class

(1) Property

Properties related to this class are defined in Table IV.84.

Table IV.84 – Property of cableCDSRecordingCollection

Integer **length**	
Type	Integer
Description	Length of cableCDSRecordingCollection

(2) Method

Methods related to this class are defined in Table IV.85.

Table IV.85 – Method of cableCDSRecordingCollection

cableCDSRecording **item** (Integer index)	
Parameters	index
Return	cableCDSRecording
Description	This method returns cableCDSRecording object at index number.

IV.6.2.7 cableCDSRecording Class

Class representing recorded content over DLNA DMS

(1) Property

Properties related to this class are defined in Table IV.86.

Table IV.86 – Property of cableCDSRecording

readonly String **objectID**	
Type	readonly String
Description	The String type property to identify the content position on the DMS. To use browseDirectChildren() method in cableCDS object, this property value should be concatenated in the tail of the parent path String.
readonly String **class**	
Type	readonly String
Description	The property to indicate the content is folder or item. String "object.container" means the content is a folder and String "object.item" means the content is an item. Item includes the Tuner content and Storaged content, and Tuner content is represented as "object.item.videoItem.videoBroadcast" or "object.item.audioItem.audioBroadcast".
readonly Boolean **restricted**	
Type	readonly Boolean
Description	The property to indicate the content can be erased or not. TRUE means that it cannot be erased.
readonly String **objectType**	
Type	readonly String
Description	The property to identify the content. "ARIB_TB" : DTT Network "ARIB_BS" : BS Network "ARIB_CS" : CS Network "CATV_CS" : Digital REMUX Network "CATV_JC" : JC-HITS TM Network

Table IV.86 – Property of cableCDSRecording

readonly String **importURI**	
Type	readonly String
Description	*URI indicating the content position on the DMS.* For example https://www.iptvserver.com：8001/Video/content.mpg? DTCPPORT1=50001
readonly String **title**	
Type	readonly String
Description	The property indicates the Title name. Equivalent to "dc:title" on DMS
readonly String **date**	
Type	readonly String
Description	The property indicates the Event start time. Equivalent to "dc:date" on DMS
readonly String **duration**	
Type	readonly String
Description	The property indicates the Event duration. Equivalent to " res@duration " on DMS
readonly String **channelName**	
Type	readonly String
Description	The property indicates the Event channel name used by service descriptor in the 2nd loop of SDT.
readonly String **channelNr**	
Type	readonly String
Description	The property indicates three number channel digits used by service list descriptor in the 2nd loop of NIT.
readonly ParentaRatingCollection **parentalRating**	
Type	readonly ParentaRatingCollection
Description	This property indicates the parental control age that the content has. The rating value in parental control descriptor is used to convert hexadecimal value as 0xXX. age = rating + 3; And if the BOX age is larger than or equal to this value, viewing is allowed. Equivalent to " upnp:rating" on DMS
readonly String **resolution**	
Type	readonly String
Description	The property indicates the resolution. The string format is, (Horizontal 4 digits) X (Vertical 4 digits). For example, "(1920) X (1024)". Equivalent to " res@ resolution" on DMS

Table IV.86 – Property of cableCDSRecording

readonly String **genre**	
Type	readonly String
Description	The property indicates the content genre. content_nibble_level_1 in EIT event loop is used and mapped according to JCTEA_STD-003. Refer to Cable SPEC-020 Annex A Equivalent to "upnp:genre" on DMS
readonly String **radioBand**	
Type	readonly String
Description	This property indicates the service belong to digital radio service if service_type is digital_radio_sound_service. If service_type is not this value, this property should not be assigned. The value should be "Satellite". Equivalent to "upnp:radioBand" on DMS
readonly BookmarkCollection **bookMarks**	
Type	readonly BookmarkCollection
Description	The property indicates the last resume point for playback. Only one Bookmark object can be used and the format of properties are, name: "UUID in cableDMS objet" time: "playPosition value for last playback"
readonly String **digitalCopyControl**	
Type	readonly String
Description	The property indicates Digital Copy Control Data. digital_recording_control_data" in digital copy control description is used. "arib@copyControlInfo" on DMS is used partially.
readonly cableCDSMetadata **cdsMetadata**	
Type	readonly cableCDSMetadata
Description	The property indicates an optional metadata properties on DMS.
readonly String **analogCopyControl**	
Type	readonly String
Description	The property indicates Digital Copy Control Data. digital_recording_control_data" in digital copy control description is used. "arib@copyControlInfo" on DMS is used partially.

IV.6.2.8 cableCDSMetadata Class

Class representing metadata as an optional property over DLNA DMS

(1) Property

Properties related to this class are defined in Table IV.87.

Table IV.87 – Property of cableCDSMetadata

readonly String **description**	
Type	readonly String
Description	This property indicates the Event short description. Short event descriptor in EIT is used.

Table IV.87 – Property of cableCDSMetadata

readonly String **longDescription**	
Type	readonly String
Description	This property indicates the Event detailed description. Extended event descriptor in EIT event is used.
readonly String **album**	
Type	readonly String
Description	This property indicates the title of the album. Equivalent to "upnp:album" on DMS
readonly String **artist**	
Type	readonly String
Description	This property indicates the name of the artist. Equivalent to "upnp:artist" on DMS
readonly String **author**	
Type	readonly String
Description	This property indicates the name of the author. Equivalent to "upnp:author" on DMS
readonly String **description**	
Type	readonly String
Description	This property indicates the Event short description. Short event descriptor in EIT is used.
readonly String **longDescription**	
Type	readonly String
Description	This property indicates the Event detailed description. Extended event descriptor in EIT event is used.
readonly String **album**	
Type	readonly String
Description	This property indicates the title of the album. Equivalent to "upnp:album" on DMS
readonly String **artist**	
Type	readonly String
Description	This property indicates the name of the artist. Equivalent to "upnp:artist" on DMS
readonly String **author**	
Type	readonly String
Description	This property indicates the name of the author. Equivalent to "upnp:author" on DMS

Appendix V

Broadcasting standards (Region A)

(This appendix does not form an integral part of this Recommendation.)

This appendix provides Region A specific issues. The STB for Region A is required to support the content defined in this appendix.

V.1 ETSI/DVB

ETSI/DVB specifications are defined in Table V.1.

Table V.1 – ETSI/DVB specifications

Spec no.	Name
[b-ETSI TS 101 154]	Specification for the use of Video and Audio Coding in Broadcasting Applications based on the MPEG-2 Transport Stream
[b-ETSI TS 102 154]	Implementation Guidelines for the use of MPEG-2 Systems, Video and Audio in Contribution Applications
[b-ETSI EN 300 429]	Framing structure, channel coding and modulation for cable systems
[b-ETSI EN 302 769]	Frame structure channel coding and modulation for a second generation digital transmission system for cable systems (DVB-C2)
[b-ETSI EN 300 468]	Specification for Service Information (SI) in DVB systems
[b-ETSI TS 101 211]	Guidelines on implementation and usage of Service Information (SI)
[b-ETSI TS 101 162]	Allocation of Service Information (SI) codes for DVB systems
[b-ETSI EN 300 743]	Subtitling systems
[b-ETSI TR 101 290]	Measurement guidelines for DVB systems
[b-ETSI TR 289]	Support for use of scrambling and Conditional Access (CA) within digital broadcasting systems
[b-ETSI TS 103 197]	Head-end implementation of DVB SimulCrypt
[b-ETSI TS 101 699]	Extensions to the Common Interface Specification
[b-ETSI ES 201 812]	Multimedia Home Platform (MHP) Specification 1.03
[b-ETSI TS 102 812]	Multimedia Home Platform (MHP) Specification 1.1
[b-ETSI TS 102 727]	Multimedia Home Platform (MHP) Specification 1.22
[b-ETSI TS 102 606]	Generic Stream Encapsulation (GSE) Protocol

V.2 IEC/CENELEC

IEC/CENELEC specifications are defined in Table V.2.

Table V.2 – IEC/CENELEC specifications

Spec no.	Name
[b-IEC/EN 60728-1]	Cable networks for television signals, sound signals and interactive services – Part 1: System performance of forward paths
[b-IEC/EN 60728-2]	Cable networks for television signals, sound signals and interactive services – Part 2: Electromagnetic compatibility for equipment
[b-IEC/EN 60728-3]	Cable networks for television signals, sound signals and interactive services – Part 3: Active wideband equipment for coaxial cable networks
[b-IEC/EN 60728-4]	Cable networks for television signals, sound signals and interactive services – Part 4: Passive wideband equipment for coaxial cable networks
[b-IEC/EN 60728-10]	Cable networks for television signals, sound signals and 1 interactive services – Part 10: System performance of return paths
[b-IEC/EN 60728-11]	Cable networks for television signals, sound signals and interactive services – Part 11: Safety
[b-IEC/EN 60728-12]	Cable networks for television signals, sound signals and interactive services – Part 8: Electromagnetic compatibility of systems

V.3 CENELEC

CENELEC specifications are defined in Table V.3.

Table V.3 – CENELEC specifications

Spec no.	Name
[b-EN 50221]	Common Interface Specification for Conditional Access and other Digital Video Broadcasting Decoder Applications
[b-R206 001]	Guidelines for implementation & use of the Common Interface for DVB Decoder Applications

V.4 ISO/IEC

ISO/IEC specifications are defined in Table V.4.

Table V.4 – ISO/IEC specifications

Spec no.	Name
[b-ISO/IEC 13818-1]	Generic coding of moving pictures and associated audio information – Part 1: Systems
[b-ISO/IEC 13818-2]	Generic coding of moving pictures and associated audio information – Part 2: Video
[b-ISO/IEC 13818-4]	Information technology – Generic coding of moving pictures and associated audio information – Part 4: Conformance testing
[b-ISO/IEC 14496-1]	Coding of audio-visual objects – Part 1: Systems
[b-ISO/IEC 14496-4]	Coding of audio-visual objects – Part 4: Conformance testing

Appendix VI

Broadcasting standards (Region B)

(This appendix does not form an integral part of this Recommendation.)

This appendix is intentionally left blank.

Appendix VII

Broadcasting standards (Region C)

(This appendix does not form an integral part of this Recommendation.)

This appendix provides Region C specific issues. The STB for Region C is required to support the content defined in this appendix.

VII.1 Broadcasting related specifications for Region C (see clause 5.3)

JCL/JLabsCable specifications are defined in Table VII.1.

Table VII.1 – JCL/JLabsCable specifications

Spec no.	Name
[b-JCL SPEC-001]	Operation Spec for the transmodulation of BS digital broadcasting
[b-JCL SPEC-001-01]	Detail Spec to protect the using without due authorization
[b-JCL SPEC-001-02]	Operation spec for down load functionality
[b-JCL SPEC-002]	Operation spec for transmodulation of east 110 degree of east longitude
[b-JCL SPEC-003]	Operation spec for remux of digital broadcasting(cable broadcasting)
[b-JCL SPEC-004]	Operation spec for remux of digital broadcasting (i-HITS)
[b-JCL SPEC-005]	Operation spec for transmodulation of JC-HITS
[b-JCL SPEC-006]	Operation spec for pass-through of DTT broadcasting
[b-JCL SPEC-007]	Operation spec for transmodulation of DTT broadcasting
[b-JCL SPEC-010]	Operation spec for interactive digital broadcasting
[b-JCL SPEC-011]	Operation spec for interactive digital cable TV(maintenance operation,web,PPV)
[b-JLabs SPEC-017]	Operation spec for high level remux digital broadcasting (Cablebroadcasting)
[b-JLabs SPEC-018	Operation spec for high level remux digital broadcasting (i-HITS)
[b-JLabs SPEC-019]	Operation spec for high level remux digital broadcasting (JC-HITS)
[b-JCL SPEC-022]	Operational specifications for ReCAS digital broadcasting

JCTEA specifications are defined in Table VII.2.

Table VII.2 – JCTEA specifications

Spec no.	Name
[b-JCTEA STD-001]	Conditional Access System for Digital Cable Television
[b-JCTEA STD-002]	Multiplex System for Digital Cable Television
[b-JCTEA STD-003]	Service Information for Digital Cable Television
[b-JCTEA STD-007]	Receiver for Digital Cable Television
[b-JCTEA STD-008]	Pass-through Method for Transmission of BS Digital Signals Over Cable Television System
[b-JCTEA STD-011]	Pass-through Method for Digital Terrestrial Television Broadcasting Signals over Cable Television System

ARIB specifications are defined in Table VII.3.

Table VII.3 – ARIB specifications

Spec no.	Name
[b-ARIB STD-B10]	Service Information for Digital Broadcasting System
[b-ARIB STD-B20]	Transmission System for Digital Satellite Broadcasting
[b-ARIB STD-B21]	Receiver for Digital Broadcasting
[b-ARIB STD-B24]	Data Coding and Transmission Specification for Digital Broadcasting
[b-ARIB STD-B25]	Conditional Access System Specifications for Digital Broadcasting
[b-ARIB STD-B31]	Transmission System for Digital Terrestrial Television Broadcasting
[b-ARIB TR-B14]	Operational Guidelines for Digital Terrestrial Television Broadcasting
[b-ARIB TR-B15]	Operational Guidelines for Digital Satellite Television Broadcasting

IPTV Forum Japan specifications are defined in Table VII.4.

Table VII.4 – IPTV Forum Japan specifications (informative)

Specification No.	Name
[b-IPTV-FJ STD-0001]	IPTV general specification
[b-IPTV-FJ STD-0002]	Specification for VOD of IPTV
[b-IPTV-FJ STD-0003]	Specification for download of IPTV
[b-IPTV-FJ STD-0004]	Specification for IP broadcasting of IPTV
[b-IPTV-FJ STD-0006]	Specification for CDN scope service approach of IPTV
[b-IPTV-FJ STD-0007]	Specification for Internet scope service approach of IPTV

Table VII.5 – Copy control and contents protection

Programs	Standards	Transmission method	Contents
BS transmodulation Remux (including community channel, i-Hits) JC-HITS transmodulation	[b-ARIB TR-B15] Part 1, vol. 2, clause 5.10	QAM	Copy control
	[b-ARIB TR-B15] Part 1, vol.8	QAM	Contents protection rule
DTT pass through, Terrestrial transmodulation	[b-ARIB TR-B14] Part 2, vol.2, clause 7.10	OFDM (pass through) QAM (transmodulation)	Copy control
	[b-ARIB TR-B14] vol. 8	OFDM (pass through) and QAM (transmodulation)	Contents protection rule

VII.2 Tuner (see clause 6.1)

The STB is required to be equipped with QAM tuners compliant with [b-JCTEA STD-007-5.0].

The STB can optionally be equipped with OFDM tuners as Non-QAM tuners compliant with [b-JCTEA STD-011-1.0].

VII.3 External interface (see clause 6.5)

The STB is required to support analogue composite output interface (e.g., RCA-pin) compliant with [b-ARIB STD-B21].

The STB is required to support content protection mechanisms compliant with [b-ARIB TR-B15] volume 8 clause 6.3 for all the output interfaces equipped with the STB.

VII.4 RCU (see clause 6.7)

The STB is recommended to support the RCU key allocation compliant with [b-JCTEA STD-007].

The IR type RCU is recommended to support IR code compliant with Japanese CE association.

VII.5 Mechanical specification (see clause 6.8)

The STB is required to be equipped with either one slot for C-CAS or two slots for B-CAS and C-CAS. The card slot is required to be compliant with [ISO7816].

VII.6 Power supply (see clause 6.11)

The connector for the power adaptor is required to be barrel plug defined by [b-JEITA RC-5320A].

VII.7 Management by SNMP (see clause 7.2.1)

The STB is required to support the following SNMP management mechanisms.

[b-JCL SPEC-011-01] version 1.0 clause 3 (when embedded CM is used)

[b-JCL SPEC-011-02] version 1.0 clause 3 (when Ethernet WAN port is used)

VII.8 STB Firmware Management (see clause 7.2.3)

The STB is required to support the BS-TS engineering download scheme defined in [b-JCL SPEC-001] for STB firmware downloading through broadcasting channel.

VII.9 Other common requirement (see clause 7.2.5)

The STB is required to support logging of broadcasting-related error code defined by [b-ARIB STD-B21].

VII.10 Receiver identification (see clause 7.3.3)

The STB is required to support 6-byte receiver ID defined in [b-JCL SPEC-001] clause 5.3.9.

VII.11 Illegal usage prevention function (see clause 7.3.4)

The STB is required to support the illegal usage prevention function defined in [b-JCL SPEC-001] clause 4.

VII.12 Service and content protection (see clause 7.3.6)

Service protection (conditional access)

The STB is required to support the specification of conditional access in Table VII.6.

Table VII.6 – Specifications of conditional access

Type of broadcasting	Standard	Transmission method
Overall (STB specification)	[b-JCTEA STD-001]	QAM and OFDM*
BS transmodulation	[b-JCL SPEC-001]	QAM
Remux (Own program)	[b-JCL SPEC-003]	QAM
Remux (i-Hits)	[b-JCL SPEC-004]	QAM
Remux (JC-HITS)	[b-JCL SPEC-005]	QAM
Terrestrial digital path through	[b-JCL SPEC-006]	OFDM*
Terrestrial digital transmodulator	[b-JCL SPEC-007]	QAM
IP broadcasting	[b-IPTVFJ STD-0004]	IP* (multicast)
* The transmission method description given for OFDM and IP shall be applied only for the STBs accommodating an OFDM tuner and/or IPTV forum signal reception function, respectively.		

The STB in Region C has two types 1) as with two slots for both B-CAS and C-CAS, and 2) with only one C-CAS slot. In case the STB has both B-CAS and C-CAS slots, selection of the CAS card shall be able to be done autonomously according to the CA_system_ID (accommodated in the PMT primary loop). Fixed allocation or distinction with other ID, as 'when network_id is 0x7F, B-CAS is used', shall not be applied. However, when the STB accommodates the OFDM tuner, it is possible for the STB to always apply B-CAS for receiving programmes by the OFDM tuner. (In this case, it also requires the autonomous selection upon receiving the QAM signal.) See Table VII.7.

Table VII.7 – Types of CAS

CA_system_ID value	CAS type
0x0003	C-CAS(1) ([b-JCL SPEC-003, 004, 005, JLabs SPEC-017, 018, 019])
0x0004	C-CAS(2) ([b-JCL SPEC-003, 004, 005, JLabs SPEC-017, 018, 019])
0x0005	B-CAS ([b-JCL SPEC-001, 006] BS-TM and DTT pass through)
0x0006	C-CAS(3) ([b-JCL SPEC-003, 004, 005, JLabs SPEC-017, 018, 019])
0x0007	B-CAS ([b-JCL SPEC-007] DTT transmodulation)
0x000D	Marlin IPTV-ES ([b-IPTVFJ STD-0004] IPTV forum IP broadcasting)

Regarding C-CAS (1) ~ (3) (0x0003, 0x0004, and 0x0006), these are uniquely decided according to the used restricted receiving facility. Although there is not a case when the changeover among three methods is being done during receiving of a programme, the STB shall be able to correspond to any one of them.

In addition, in the case where only one C-CAS slot is installed, C-CAS card shall be applied to any of the CA_system_ID value (ox0003 ~ 0x0007). In this case, even for the programme to which the B-CAS is applied in nature, the cable NW operator shall apply any measure to make the C-CAS card applicable (Alteration of ECM, or sending Kw to C-CAS card with EMM).

Content protection

The STB is required to support copy control and content protection rules compliant with the specifications in Table VII.8.

Table VII.8 – Copy control and contents protection

Programs	Standards	Transmission method	Contents
BS transmodulation Remux (Own i-Hits) JC-HITS transmodulation	[b-ARIB TR-B15] Part 1, vol. 2, clause 5.10	QAM	Copy control
	[b-ARIB TR-B15] Part 1, vol.8	QAM	Contents protection rule
DTT pass through, Terrestrial transmodulation	[b-ARIB TR-B14] Part 2, vol.2, clause 7.10	OFDM (pass through) QAM (transmodulation)	Copy control
	[b-ARIB TR-B14] vol. 8	OFDM (pass through) and QAM (transmodulation)	Contents protection rule

VII.13 DVR and relating function (see clause 7.8)

The definition of the term recoding is required to follow the [b-ARIB TR-B15] volume 8 clause 4.

The STB is required to support the content storing functionality compliant with [b-ARIB TR-B15] volume 8 clause 6.5.

The STB is required to support content recording to internal HDD or external USB HDD compliant with [b-ARIB TR-B15] volume 8 clause 7.2.4.

If the STB is equipped with a recording capability to removable media, the STB is required to be compliant with [b-ARIB TR-B15] volume 8 Appendix B.

VII.14 Decoder for broadcasting type service (clause 8.1)

The STB is required to support a media decoder for broadcasting services compliant with [b-JCTEA STD-007] with [b-ARIB STD-B1] and [b-ARIB STD-B21, b-ARIB STD-B10, b-ARIB STD-B20 and b-ARIB TR-B15].

Video decoder

– The STB is required to support [b-JLabs SPEC-017] ver.1.0 clause 8.5.3 "Operation spec for advanced remux of digital broadcasting" (cable broadcasting). Although this specification defines VOD as an optional functionality, the STB defined in this Recommendation is required to support VOD as mandatory.

Audio decoder

– The STB is required to support [b-JCTEA STD-007] clauses 5.1 and 6.2.

VII.15 3D content identification (see clause 8.4)

For the STB for Region C, this clause overrides clause 8.4 as follows.

The STB is required to support the 3D content identification methods available in Table VII.9.

Table VII.9 – Identification method of 3D content

3D Picture format	Compress method	3D identification	Reference
Side-by-Side	MPEG-2	Utilize user_data in ES	[b-ARIB TR-B15] (Part1 Vol.7 clause 7.9)
	[ITU-T H.264]	Frame Packing Arrangement SEI	[ITU-T H.264] Annex D (D.1.25) and ARIB TR-B15(Part1 Vol. 7 clause 10.1
Full HD (temporal interleaving frame arrangement)	[ITU-T H.264] MVC	Frame Packing Arrangement SEI	[ITU-T H.264] Annex H
NOTE – 3D signalling which place L and R frames alternatively is so called "Frame Sequential" but in ITU-T Recommendation [ITU-T H.264], it is described as "a temporal interleaving frame arrangement" and thus this specification uses this term.			

The STB is required to support the following issues for playback of 3D content.

In the case of MPEG2 playback, ARIB TR-B15 Part1 vol.7 clause 7.9 describing 3D identification should be used.

For [ITU-T H.264] 3D content, the STB is required to support all the parameters defined in [b-ARIB TR-B15] Part1 Vol.7 clause 10.1, and additionally support 3D content with the parameter of frame_packing_arrangement_type = 5 (temporal interleave frame arrangement) in Frame Packing Arrangement SEI.

Both for MPEG2 and [ITU-T H.264], the playable screen size, bit rate condition follows the above specification's profile and level.

Bibliography

[b-ITU-T H.741.1] Recommendation ITU-T H.741.1 (2012), *IPTV application event handling: Audience measurement operations for IPTV services*.

[b-ITU-T H.741.2] Recommendation ITU-T H.741.2 (2012), *IPTV application event handling: Data structures of audience measurement for IPTV services*.

[b-ITU-T H.741.3] Recommendation ITU-T H.741.3 (2012), *IPTV application event handling: Audience measurement for IPTV distributed content services*.

[b-ETSI EN 300 429] ETSI EN 300 429 (1998), *Digital Video Broadcasting (DVB) – Framing structure, channel coding and modulation for cable systems*.

[b-ETSI EN 300 468] ETSI EN 300 468 (2009), *Digital Video Broadcasting (DVB) – Specification for Service Information (SI) in DVB systems*.

[b-ETSI EN 300 743] ETSI EN 300 743 (2006), *Digital Video Broadcasting (DVB) – Subtitling systems*.

[b-ETSI EN 302 769] ETSI EN 302 769 (2010), *Digital Video Broadcasting (DVB) – Frame structure channel coding and modulation for a second generation digital transmission system for cable systems (DVB-C2)*.

[b-ETSI ES 201 812] ETSI ES 201 812 V1.1.2 (2006), *Digital Video Broadcasting (DVB) – Multimedia Home Platform (MHP) Specification 1.03*.

[b-ETSI TR 101 290] ETSI TR 101 290 (2001), *Digital Video Broadcasting (DVB) – Measurement guidelines for DVB systems*.

[b-ETSI TR 289] ETSI TR 289 (1996), *Digital Video Broadcasting (DVB) – Support for use of scrambling and Conditional Access (CA) within digital broadcasting systems*.

[b-ETSI TS 101 154] ETSI TS 101 154 (2009), *Digital Video Broadcasting (DVB) – Specification for the use of Video and Audio Coding in Broadcasting Applications based on the MPEG-2 Transport Stream*.

[b-ETSI TS 101 162] ETSI TS 101 162 (2009), *Digital Video Broadcasting (DVB) – Allocation of Service Information (SI) and Data Broadcasting Codes for Digital Video Broadcasting (DVB) systems*.

[b-ETSI TS 101 211] ETSI TS 101 211 (2012), *Digital Video Broadcasting (DVB) – Guidelines on implementation and usage of Service Information (SI)*.

[b-ETSI TS 101 699] ETSI TS 101 699 (1999), *Digital Video Broadcasting (DVB) – Extensions to the Common Interface Specification*.

[b-ETSI TS 102 154] ETSI TS 102 154 (2004), *Digital Video Broadcasting (DVB) – Implementation guidelines for the use of Video and Audio Coding in Contribution and Primary Distribution Applications based on the MPEG-2 Transport Stream.*

[b-ETSI TS 102 606] ETSI TS 102 606 V1.1.1 (2007), *Digital Video Broadcasting (DVB) – Generic Stream Encapsulation (GSE) Protocol*.

[b-ETSI TS 102 727] ETSI TS 102 727 V1.1.1 (2007), *Digital Video Broadcasting (DVB) – Multimedia Home Platform (MHP) Specification 1.22*.

[b-ETSI TS 102 812] ETSI TS 102 812 V1.1.1 (2001), *Digital Video Broadcasting (DVB) – Multimedia Home Platform (MHP) Specification 1.1*.

[b-ETSI TS 103 197] ETSI TS 103 197 (2008), *Digital Video Broadcasting (DVB) – Head-end implementation of DVB SimulCrypt*.

[b-IEC 60068-1] IEC 60068-1:1988, *Environmental testing – Part 1: General and guidance*.

[b-IEC 60068-2-1] IEC 60068-2-1:2007, *Environmental testing – Part 2: Tests. Test A: Cold*.

[b-IEC 60068-2-2] IEC 60068-2-2:2007, *Environmental testing – Part 2: Tests. Test B: Dry heat*.

[b-IEC 60068-2-6] IEC 60068-2-6:2007, *Environmental testing – Part 2: Tests – Test Fc: Vibration (sinusoidal).*

[b-IEC 60068-2-14] IEC 60068-2-14:2009, *Environmental testing – Part 2: Tests. Test N: Change of temperature*.

[b-IEC 60068-2-27] IEC 60068-2-27:2008, *Environmental testing – Part 2: Tests. Test Ea and guidance: Shock*.

[b-IEC 60068-2-30] IEC 60068-2-30:2005, *Environmental testing – Part 2: Tests. Test Db and guidance: Damp heat, cyclic (12 + 12-hour cycle).*

[b-IEC 60068-2-31] IEC 60068-2-31:2008 *Environmental testing – Part 2-31: Tests – Test Ec: Rough handling shocks, primarily for equipment-type specimens*.

[b-IEC 60068-2-40] IEC 60068-2-40:1976, *Environmental testing – Part 2: Tests. Test Z/AM: Combined cold/low air pressure tests*.

[b-IEC/EN 50221] IEC/EN 50221 (1997), *Common Interface Specification for Conditional Access and other Digital Video Broadcasting Decoder Applications*.

[b-IEC/EN 60728-1] IEC 60728-1 ed4.0 (2007), *Cable networks for television signals, sound signals and interactive services – Part 1: System performance of forward paths*.

[b-IEC/EN 60728-2] IEC 60728-2 ed2.0 (2010), *Cable networks for television signals, sound signals and interactive services – Part 2: Electromagnetic compatibility for equipment*.

[b-IEC/EN 60728-3] IEC 60728-3 ed4.0 (2010), *Cable networks for television signals, sound signals and interactive services – Part 3: Active wideband equipment for cable networks*.

[b-IEC/EN 60728-4] IEC 60728-4 ed3.0 (2007), *Cable networks for television signals, sound signals and interactive services – Part 4: Passive wideband equipment for coaxial cable networks*.

[b-IEC/EN 60728-10] IEC 60728-10 ed2.0 (2005), *Cable networks for television signals, sound signals and 1 interactive services – Part 10: System performance of return paths*.

[b-IEC/EN 60728-11] IEC 60728-11 ed3.0 (2010), *Cable networks for television signals, sound signals and interactive services – Part 11: Safety*.

[b-IEC/EN 60728-12] IEC 60728-12 ed1.0 (2001), *Cable networks for television signals, sound signals and interactive services – Part 8: Electromagnetic compatibility of systems*.

[b-IETF RFC 768] IETF RFC 768 (1990), *User Datagram Protocol.*

[b-ISO/IEC 13818-1] ISO/IEC 13818-1 ed3.0:2007, *Information technology – Generic coding of moving pictures and associated audio information – Part 1: Systems*.

[b-ISO/IEC 13818-2] ISO/IEC 13818-2:2000, *Information technology – Generic coding of moving pictures and associated audio information – Part 2: Video*.

[b-ISO/IEC 13818-4] ISO/IEC 13818-4:2004, *Information technology – Generic coding of moving pictures and associated audio information – Part 4: Conformance testing*.

[b-ISO/IEC 14496-1] ISO/IEC 14496-1:2010, *Information technology – Coding of audio-visual objects – Part 1: Systems*.

[b-ISO/IEC 14496-4] ISO/IEC 14496-4:2004, *Information technology – Coding of audio-visual objects – Part 4: Conformance testing*.

[b-ARIB STD-B10] ARIB STD-B10 Version 4.8 (2010), *Service information for digital broadcasting system*.

[b-ARIB STD-B20] ARIB STD-B20 Version 3.0(2001), *Transmission system for digital satellite broadcasting*.

[b-ARIB STD-B21] ARIB STD-B21 Version 5.0 (2010/7), *Receiver for digital broadcasting*.

[b-ARIB STD-B24] ARIB STD-B24 Version 5.4 (2009), *Data coding and transmission specification for digital broadcasting*.

[b-ARIB STD-B25] ARIB STD-B25 Version 5.1 (2009), *Conditional access system specifications for digital broadcasting*.

[b-ARIB STD-B31] ARIB STD-B31 Version 1.9 (2010), *Transmission system for digital terrestrial television broadcasting*.

[b-ARIB STD-B32] ARIB STD-B32 Version 2.4 (2010), *Video coding, audio coding and multiplexing specifications for digital broadcasting*.

[b-ARIB TR-B14] ARIB TR-B14 Version 4.3 (2010), *Operational guidelines for digital terrestrial television broadcasting*.

[b-ARIB TR-B15] ARIB TR-B15 Version 5.1 (2010), *Operational guidelines for digital satellite broadcasting*.

[b-DAVIC 1.4] Digital Audio-Visual Council, (DAVIC) 1.4 Part 9 (1998*), Annex B, the file format of audio encoding of PCM.*.

[b-IPTV-FJ STD-0001] IPTV Forum Japan IPTV-FJ STD-0001 (2009), *IPTV general specification*.
https://www.iptvforum.jp/download/CDNScope13e.pdf

[b-IPTV-FJ STD-0002] IPTV Forum Japan IPTV-FJ STD-0002 (2010), *Specification for VOD of IPTV*.
https://www.iptvforum.jp/download/VOD11e.pdf

[b-IPTV-FJ STD-0003] IPTV Forum Japan IPTV-FJ STD-0003 (2009), *Specification for download of IPTV*.
https://www.iptvforum.jp/en/download/input.html

[b-IPTV-FJ STD-0004] IPTV Forum Japan IPTV-FJ STD-0004 (2012), *Specification for IP broadcasting of IPTV*.
https://www.iptvforum.jp/en/download/input.html

[b-IPTV-FJ STD-0006] IPTV Forum Japan IPTV-FJ STD-0006 (2010), *Specification for CDN scope service approach of IPTV*.
https://www.iptvforum.jp/en/download/input.html

[b-IPTV-FJ STD-0007] IPTV Forum Japan IPTV-FJ STD-0007 (2012), *Specification for Internet scope service approach of IPTV*. https://www.iptvforum.jp/en/download/input.html

[b-JCL SPEC-001] JCL SPEC-001 Version 1.01 (2008), *Operational specifications of trans-modulation for BS digital Satellite broadcasting*.

[b-JCL SPEC-001-01] JCL SPEC-001-01 Version 1.0 (2009), *Detail Spec to protect the using without due authorization*.

[b-JCL SPEC-001-02] JCL SPEC-001-02 Version 1.0 (2009), *Operation spec for download functionality*.

[b-JCL SPEC-002] JCL SPEC-002 Version 1.01 (2008), *Operational specifications of trans-modulation for 110 degree east longitude CS digital satellite broadcasting*.

[b-JCL SPEC-003] JCL SPEC-003 Version 2.1 (2010), *Operational specifications for remux digital broadcasting (Independent broadcasting)*.

[b-JCL SPEC-004] JCL SPEC-004 Version 2.1 (2010), *Operational specifications for remux digital broadcasting(i-HITS)*.

[b-JCL SPEC-005] JCL SPEC-005 Version 2.2 (2010), *Operational specifications of transmodulation for JC-HITS*.

[b-JCL SPEC-006] JCL SPEC-006 Version 2.01 (2008), *Operational specifications of pass through for digital terrestrial television broadcasting*.

[b-JCL SPEC-007] JCL SPEC-007 Version 2.01 (2008), *Operational specifications of trans-modulation for digital terrestrial television broadcasting*.

[b-JCL SPEC-010] JCL SPEC-010 Version 1.0 (2003), *Operational specifications for interactive digital broadcasting*.

[b-JCL SPEC-011] JCL SPEC-011 Version 1.0 (2005/12), *Operational specifications for digital interactive cable television*.

[b-JCTEA STD-001] JCTEA STD-001-1.1(2008), *Conditional Access System for Digital Cable Television*.

[b-JCTEA STD-002] JCTEA STD-002-5.1 (2007), *Multiplex System for Digital Cable Television*.

[b-JCTEA STD-003] JCTEA STD-003-5.1 (2008), *Service Information for Digital Cable Television*.

[b-JCTEA STD-007] JCTEA STD-007-5.2 (2007), *Receiver for Digital Cable Television*.

[b-JCTEA STD-008] JCTEA STD-008-1.1 (2006), *Pass-through Method for Transmission of BS Digital Signals Over Cable Television System*.

[b-JCTEA STD-011] JCTEA STD-011-1.0 (2003), *Pass-through Method for Digital Terrestrial Television Broadcasting Signals over Cable Television System*.

[b-JEITA RC-5320A] Japan Electronics and Information Technology Industries Association/JEITA RC-5320A standard (1989/1992), *DC Power Supply Plugs / Jacks*.

[b-JLabs SPEC-017] JLabs SPEC-017 (2011), *Operational specifications for advanced remux digital broadcasting (Independent broadcasting)*.

[b-JLabs SPEC-018] JLabs SPEC-018 (2011), *Operational specifications for advanced remux digital broadcasting (i-HITS).*

[b-JLabs SPEC-019] JLabs SPEC-019 (2011), *Operational specifications for advanced remux digital broadcasting (JC-HITS).*

[b-JLabs SPEC-020] JLabs SPEC-020 (2011), *Operational specifications for recording and playback technology DLNA.*

[b-JLabs SPEC-022] JLabs SPEC-022, *Operational specifications for ReCAS digital broadcasting*.

[b-JLabs DOC-009] Jlabs DOC-009-00-1.0 (2010), *Cable Access System for Digital Cable Television*.

[b-OIPF DAE] *Open* IPTV Forum (OIPF) Release 2 (2010), *Specification Declarative Application Environment.*
http://www.oipf.tv

[b-R206 001] R206 001 (1998), technical report prepared by the Technical Committee CENELEC TC 206: *Guidelines for implementation & use of the Common Interface for DVB Decoder Applications*.
http://www.dvb.org/technology/standards/R206-001.V1.pdf

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	**Cable networks and transmission of television, sound programme and other multimedia signals**
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Terminals and subjective and objective assessment methods
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	Global information infrastructure, Internet protocol aspects and next-generation networks
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2013