ETSI TS 102 824 V1.2.1 (2010-02)

Technical Specification

Digital Video Broadcasting (DVB); Remote Management and Firmware Update System for DVB IPTV Services (Phase 2)

Reference
RTS/JTC-DVB-272

Keywords
broadcasting, digital, DVB, IP

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, please send your comment to one of the following services:
http://portal.etsi.org/chaircor/ETSI_support.asp

Copyright Notification

Contents

Intellectual Property Rights 6
Foreword 6
1 Scope 7
2 References 8
2.1 Normative references 8
2.2 Informative references 10
3 Definitions and abbreviations 10
3.1 Definitions 10
3.2 Abbreviations 11
3.3 Notations 12
3.3.1 Augmented Backus-Naur Form (ABNF) 12
4 Reference Model 13
4.1 Modes of operation 13
4.1.1 RMS-only mode 14
4.1.2 RMS-FUS mode 14
4.1.3 FUS-only mode 15
4.2 Type of firmware update files 15
4.3 The role of the entities in the RMS-FUS architecture 16
4.3.1 CPEs 16
4.3.2 RMS Administrator 16
4.3.3 RMS 16
4.3.4 CE manufacturer 17
4.3.5 FUS 17
4.4 Control priorities of RMS vs. FUS-only 18
4.5 CPE Bootstrap process 18
4.5.1 FUSS Process (Managed and unmanaged CPEs) 18
4.5.2 Process after FUSS (Managed CPEs) 19
4.6 Running state behaviour of CPEs 19
4.6.1 Unmanaged CPEs 19
4.6.2 Managed CPEs 19
5 The RMS-FUS sub-systems 20
5.1 Interfaces overview 20
5.2 The Remote Management System (RMS) 22
5.2.1 Communication with RMS administration 22
5.2.2 Management of the inventory 22
5.2.3 CPE management interface 22
5.2.4 Processing of metadata and communication with the FUS (interface 4) 23
5.3 The Firmware Update System (FUS) 23
5.3.1 FUS manager and storage modules 23
5.3.2 Signalling and messaging to enable CPEs to locate updates 24
5.3.2.1 Multicast announcement service 25
5.3.2.2 Unicast query/response channel (QRC) 25
5.3.3 Firmware update delivery 25
5.3.3.1 Delivery failure in a managed environment 25
5.3.3.2 Delivery failure in an unmanaged environment 25
5.4 Secure Message Exchange and Secure Network Time 25
5.4.1 Secure Message Exchange 26
5.4.2 Secure Network Time 26
6 The interfaces 26
6.1 Interface 1 - Firmware Update file from CE manufacturer to FUS manager 27
6.1.1 Payload coding and format 28
6.2 Interface 2 - Associated metadata from CE manufacturer to FUS manager 28
6.3 Interface 3 - CE manufacturer to RMS 28

6.4 Interface 4 - RMS-FUS control interface 28
6.5 Interface 5 - Multicast delivery of firmware update file to network 29
6.5.1 Payload coding and format 29
6.5.2 Security 29
6.5.3 Delivery protocol 29
6.5.3.1 FLUTE 29
6.5.3.1.1 Profiling of FLUTE file download mechanism in DVB IPI Firmware Update Service 29
6.5.3.1.2 FLUTE session description in DVB IPI Firmware Update Service 30
6.5.3.2 DSM-CC 30
6.6 Interface 6 - Unicast delivery of firmware update file to network 31
6.6.1 Payload coding and format 31
6.6.2 Security 31
6.6.2.1 HyperText Transport Protocol (HTTP) 31
6.6.2.2 File Transfer Protocol (FTP) 31
6.6.2.3 Trivial File Transfer Protocol (TFTP) 31
6.6.3 Delivery protocol 32
6.7 Interface 7 - Firmware Update announcement service 32
6.7.1 Download discovery navigation using the multicast announcement message service 32
6.7.2 Carriage of multicast announcement over SAP transport 36
6.7.2.1 Coding of Session Announcement Protocol 36
6.7.2.1.1 Limitations 36
6.7.2.1.2 Authentication method to be used in SAP payload 37
6.7.2.2 Payload coding of SDP header 37
6.7.2.3 Coding of media sections 39
6.7.2.4 IP address range options 39
6.7.2.5 Security 39
6.7.3 Carriage over DVBSTP 39
6.7.3.1 Coding of DVBSTP 40
6.7.3.2 Payload coding for DVBSTP 41
6.7.3.3 IP address options for DVBSTP 41
6.7.3.4 Security 41
6.8 Interface 8 - Query/response interface from FUS to home environment (QRC) 41
6.8.1 Download discovery navigation using the unicast query/response channel 42
6.8.2 Payload coding and format 42
6.8.2.1 FirmwareUpdateCheck/FirmwareUpdateCheckResponse 42
6.8.3 Security 44
6.8.4 Delivery protocol 44
6.8.5 Message back-off and retry strategy 45
6.9 Interface 9 - CPE management interface 45
6.9.1 Management interface requirements 45
6.9.2 Security 45
6.9.3 Delivery protocol 45
6.10 Interface 10 - RMS management and control 45
6.11 Interface 11 - FUS control interface and file management interface 45
6.12 Interface 12 - Update file streaming to delivery formatter 45
6.13 Interface 13 - RMS device registration 46

7 Data Model 46

Annex A (normative): Use of TR-069 / CWMP for DVB RMS-FUS 47

A.1 Reference specifications within the TR-069 framework 47

A.2 CWMP Remote Procedure Calls 47
A.2.1 RPCs required at the RMS and CPE side by TR-069 Framework 48
A.2.2 RPCs optional at the RMS and CPE side by TR-069 Framework 48

A.3 Remote Management functionality using CWMP 49
A.3.1 How to exploit CWMP for DVB remote management functions 49
A.3.2 Examples of firmware update tasks in a managed environment 50

Annex B (informative): Metadata schema 52

Annex C (normative): Uniform Resource Identifier (URI) scheme for the connection to a multicast service 65

C.1 Basic DVB-MCAST URI scheme 65

C.2 RMS-FUS usage of DVB-MCAST URI scheme for DVBSTP 65

C.3 RMS-FUS usage of DVB-MCAST URI scheme for SAP 66

C.4 RMS-FUS usage of DVB-MCAST URI scheme for DSM-CC 66

C.5 RMS-FUS usage of DVB-MCAST URI scheme for FLUTE 66

Annex D (normative): SDP attributes specifically defined for DVB RMS 68

D.1 FLUTE specific attributes (multicast update announcement) 69

D.2 DSM-CC specific attributes (multicast update announcement) 70

D.3 SAP media type usage (pointer announcement) 70

D.4 DVBSTP media type usage (pointer announcement) 71

D.5 Query announcement 72

D.6 Unicast announcement 73

D.7 Target filtering attributes 73

D.7.1 CE manufacturer filter 73

D.7.2 Device class filter 74

D.7.3 Device group filter 74

Annex E (normative): DVB specific data model extensions required for DVB Content Download Service 76

E.1 CDS Push Service storage management 76

E.2 CDS Pull Service storage management 76

Annex F (normative): Extension of TR-069 / CWMP to carry DVB specific Content Download Service data model extensions 78

F.1 Rationale for the chosen solution and guidelines for implementation 79

F.2 DVB data model extensions 81

F.3 Remote Management functionality using CWMP 82

History 83

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Specification (TS) has been produced by Joint Technical Committee (JTC) Broadcast of the European Broadcasting Union (EBU), Comité Européen de Normalisation ELECtrotechnique (CENELEC) and the European Telecommunications Standards Institute (ETSI).

NOTE: The EBU/ETSI JTC Broadcast was established in 1990 to co-ordinate the drafting of standards in the specific field of broadcasting and related fields. Since 1995 the JTC Broadcast became a tripartite body by including in the Memorandum of Understanding also CENELEC, which is responsible for the standardization of radio and television receivers. The EBU is a professional association of broadcasting organizations whose work includes the co-ordination of its members' activities in the technical, legal, programme-making and programme-exchange domains. The EBU has active members in about 60 countries in the European broadcasting area; its headquarters is in Geneva.

European Broadcasting Union
CH-1218 GRAND SACONNEX (Geneva)
Switzerland
Tel: +41 22 717 21 11
Fax: +41 22 717 24 81

The Digital Video Broadcasting Project (DVB) is an industry-led consortium of broadcasters, manufacturers, network operators, software developers, regulatory bodies, content owners and others committed to designing global standards for the delivery of digital television and data services. DVB fosters market driven solutions that meet the needs and economic circumstances of broadcast industry stakeholders and consumers. DVB standards cover all aspects of digital television from transmission through interfacing, conditional access and interactivity for digital video, audio and data. The consortium came together in 1993 to provide global standardisation, interoperability and future proof specifications.

1 Scope

The present document contains the specification for the remote management and firmware update system for DVB IPTV services and forms an addendum to the DVB IPTV Handbook TS 102 034 [1]. All aspects of the RMS and FUS functionality which are standardised by DVB are described within this single common document, but since the RMS and associated FUS service is specified to be directly associated with the DVB IPTV Handbook TS 102 034 [1] some direct references to parts of that specification are included. The association with [1] does not prevent the present document partially or completely being used in conjunction with other IPTV delivery services.

Remote Management is the ability of a server entity outside the home environment to monitor and configure the devices within the home and covers provisioning and assurance tasks. It optionally includes firmware updates to the equipment.

The remote management entity is referred to as the Remote Management System (RMS) and the firmware update capability as the Firmware Update System (FUS).

This updated version of the document also now includes:

- Addition of specific RMS-FUS payloadID for DVBSTP announcement messages (0xB2).
- The definition of the data model extension objects to support the DVB Content Delivey System specific functionality, intended to be used in conjunction with the data models for IPTV CPEs defined by other organisation, e.g. Broadband Forum, Cablelabs.
- An annex defining the way in which the DVB specific data model objects can work within the Broadband Forum data model. Similar modelling could be for data models from other standards organisations but that has not been done in the present document.
- DVB RMS-FUS profiling of the authentication method specified for the SAP multicast protocol for carrying the firmware update announcement messages.
- An extension to the metadata which can support usage of multiple file objects in a single carousel (FLUTE or DSM-CC), and multiple DSM-CC data components within a service for firmware update delivery.
- Profiling of the dvb-mcast URI specific to each delivery protocol, allowing support for usage of multiple file objects in a single delivery carousel (FLUTE or DSM-CC), multiple DSM-CC data components within a service for firmware update delivery and multiple channels for FLUTE.

The present document is intended to provide additional functionality that can be used in conjunction with existing RMS specifications standardised by other bodies and primarily focuses on defining a multicast firmware update capability. A description of the additional functionality for an RMS interface based on the Broadband Forum TR-069 framework is presented in annex A.

The targeting of the firmware updates may include all the devices in the home environment if the CE manufacturer supplies updates, and although we are at present looking only at IPTV delivery we should consider that the solution does not preclude delivery to home network devices which would potentially be "hidden" behind a gateway device in future phases.

The term "CPE" has been adopted in the present document in preference to the more well-known DVB-IPI terms "HNED" (Home Network End Device) and "DNG" (Delivery Network Gateway) in order to take other DVB devices such as the home network devices currently being specified by DVB-IPI HN task force into account Both DNG and HNED are included in the scope of CPE in this context.

The logical architecture required to realise the RMS and FUS environments is described in the present document. The RMS and FUS may be provided by the same or different agencies, and may be co-located. The entities identified as building blocks of RMS and FUS are logical entities rather than physical devices, and a single manufacturer could integrate multiple logical entities into a single physical device.

The RMS and the FUS service may be provided by the service provider, network operator or a third party on an agreed contract basis.

Note that there may be several CE manufacturers contributing firmware files, and several FUSs, which may be managed by one or more RMSs (but with only one active management service at a time for any single CPE) delivered through multiple networks to the various populations of CPEs. However, some rules on prioritisation in terms of CPE management are defined within the present document.

The options for delivery of the update files over the IPTV delivery network to the CPEs are specified in the present document in the following clauses.

Security is an important aspect of all communications in an IP environment and the requirements for and methods of securely either exchanging messaging or sending firmware update payloads are described in each clause of the present document where it is possible and in a way appropriate to the part of the specification. These are referenced to a central clause describing the principles used within the present document. However, it is not intended for carrying the most secure downloads, such as those where DRM related or contractual conditions apply.

Packaging and authentication of firmware updates delivered over multicast DVBSTP is not specified in the present document, it is assumed that those functions will be defined at a manufacturer-specific level.

2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific.

- For a specific reference, subsequent revisions do not apply.
- Non-specific reference may be made only to a complete document or a part thereof and only in the following cases:
  - if it is accepted that it will be possible to use all future changes of the referenced document for the purposes of the referring document;
  - for informative references.

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

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

2.1 Normative references

The following referenced documents are indispensable for the application of the present document. For dated references, only the edition cited applies. For non-specific references, the latest edition of the referenced document (including any amendments) applies.

[1] ETSI TS 102 034 (V1.4.1): "Digital Video Broadcasting (DVB); Transport of MPEG-2 TS Based DVB Services over IP Based Networks".

[2] ETSI TS 102 006 (V1.3.1): "Digital Video Broadcasting (DVB); Specification for System Software Update in DVB Systems".

[3] ISO/IEC 13818-6 (1998): "Information technology - Generic coding of moving pictures and associated audio information - Part 6: Extensions for DSM-CC" .

[4] ETSI EN 301 192: "Digital Video Broadcasting (DVB); DVB specification for data broadcasting".

[5] Broadband Forum TR-140: "TR-069 Data Model for Storage Service Enabled Devices", Issue Number 1.1, December 2007.

NOTE: Available at http://www.broadband-forum.org/technical/download/TR-140_Issue1.1.pdf.

[6] Broadband Forum TR-069 Amendment 2: "CPE WAN Management Protocol", December 2007.

NOTE: Available at http://www.broadband-forum.org/technical/download/TR-069.pdf and in the context of the present document it is referred to generally as "TR-069" or "TR-069 Framework".

[7] Broadband Forum TR-106: "Data Model Template for TR-069-Enabled Devices", September 2009.

NOTE: Available at http://www.broadband-forum.org/technical/download/TR-106_Amendment-3.pdf.

[8] Broadband Forum TR-135: "Data model for a TR-069 enabled STB", December 2007.

NOTE: Available at http://www.broadband-forum.org/technical/download/TR-135.pdf.

[9] Broadband Forum TR-157: "Component Objects for CWMP", March 2009.

NOTE: Available at http://www.broadband-forum.org/technical/download/TR-157.pdf.

[10] Web pages of IEEE: "Organizationally Unique Identifiers (OUIs)".

NOTE: Available at http://standards.ieee.org/faqs/OUI.html.

[11] W3C: "Extensible Markup Language (XML) 1.0 (Second Edition)".

NOTE: Available at http://www.w3.org/TR/2000/REC-xml-20001006.

[12] W3C: "Simple Object Access Protocol (SOAP) 1.1".

NOTE: Available at http://www.w3.org/TR/2000/NOTE-SOAP-20000508.

[13] IETF RFC 3268: "Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)".

[14] IETF RFC 2616: "Hypertext Transfer Protocol - HTTP/1.1".

[15] IETF RFC 4346: "The Transport Layer Security (TLS) Protocol Version 1.1".

[16] IETF RFC 2246: "The Transport Layer Security (TLS) Protocol, Version 1.0".

[17] IETF RFC 1305: "Network Time Protocol (Version 3) Specification, Implementation and Analysis".

[18] IETF RFC 1321: "The MD5 Message-Digest Algorithm".

[19] IETF RFC 1112: "Host extensions for IP multicasting".

[20] IETF RFC 3926: "FLUTE - File Delivery over Unidirectional Transport".

[21] IETF RFC 3450: "Asynchronous Layered Coding (ALC) Protocol Instantiation".

[22] IETF RFC 3451: "Layered Coding Transport (LCT) Building Block".

[23] IETF RFC 3452: "Forward Error Correction (FEC) Building Block".

[24] IETF RFC 1952: "GZIP file format specification version 4.3".

[25] IETF RFC 1812: "Requirements for IP Version 4 Routers".

[26] IETF RFC 4566: "SDP: Session Description Protocol".

[27] IETF RFC 2974: "Session Announcement Protocol".

[28] IETF RFC 2234: "Augmented BNF for Syntax Specifications: ABNF".

[29] IETF RFC 2236: "Internet Group Management Protocol, Version 2".

[30] IETF RFC 3376: "Internet Group Management Protocol, Version 3".

[31] IETF RFC 3986: "Uniform Resource Identifier (URI): Generic Syntax".

[32] IETF RFC 3852: "Cryptographic Message Syntax (CMS)".

[33] Void.

[34] IETF RFC 2818: "HTTP Over TLS".

[35] IETF RFC 4217: "Securing FTP with TLS".

[36] IETF RFC 4607: "Source-Specific Multicast for IP".

[37] ETSI TS 102 472: "Digital Video Broadcasting (DVB); IP Datacast over DVB-H: Content Delivery Protocols".

[38] ETSI TS 102 851: "Digital Video Broadcasting (DVB); Uniform Resource Identifiers (URI) for DVB Systems".

[39] ETSI TS 102 822-2: "Broadcast and On-line Services: Search, select, and rightful use of content on personal storage systems ("TV-Anytime"); Part 2: Phase 1 - System description".

[40] Web Services Interoperability Organisation: "Basic Security Profile 1.0".

NOTE: Available at http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html.

[41] Web Services Interoperability Organisation: "Basic Profile 1.0".

NOTE: Available at http://www.ws-i.org/Profiles/BasicProfile-1.0.html.

2.2 Informative references

The following referenced documents are not essential to the use of the present document but they assist the user with regard to a particular subject area. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Not applicable.

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

NOTE: The following definitions for entities, services and functionalities described in the present document are given on an informative basis, with the IPTV Handbook TS 102 034 [1] containing the normative versions of these definitions.

ACS: Broadband Forum description for the server offering Remote Management functionality

boot process: sequence necessary to provision the CPE in terms of the entry IP addresses for the RMS, FUS and other DVB services

CE manufacturer: context of the present document this refers to the agent responsible for delivering the firmware update image file and the associated metadata to the FUS and RMS as appropriate

NOTE: In the context of the present document it may also be an alternative agency, other than the actual CE manufacturer, who supplies the firmware update and the metadata.

CPE: generic method used in the context of this present document to refer collectively to HNEDs and DNGs

Delivery Network: connection into the home between the delivery network gateway and the service provider

Delivery Network Gateway (DNG): device referred to in TS 102 034 [1] which is connected to one or multiple delivery networks and one or multiple home network segments

NOTE: See TS 102 034 [1].

DVB-IP service: DVB service provided over IP or content on demand over IP

firmware: system software of the CPE

Home Network End Device (HNED): item of equipment which is connected to a home network and which typically terminates the IP based information flow (sender or receiver side)

NOTE: See TS 102 034 [1].

managed CPE: device in the home which is managed by RMS in managed network

packaging: processing of a file or object in preparation for distribution over the network

pointer announcement: information carried over the multicast service carrying redirection pointers to other locations where additional pointer, update, query, or unicast announcements are available

query announcement: information carried over the multicast service carrying redirection pointers to the location where connection can be made to the query/response channel

Remote Management System (RMS): server used for DVB Remote Management functionality

NOTE: Where BBF TR-069 methods are used this functionality is similar to the ACS in BBF terminology for the services specified by DVB.

Service Provider (SP): entity providing a service to the end-user

NOTE: In the context of the present document, SP will mean a Service Provider providing DVB-IP services.

Set Top Box (STB): method of referring to a physical device in the home, equivalent to "CPE" in the context of the present document

unicast announcement: information carried over the multicast service carrying redirection pointers to the location where connection can be made to download an update using a unicast service without any further navigation

unmanaged CPE: device in the home which is not managed by RMS or any other entities in managed network

update announcement: information carried over the multicast service carrying the descriptive information about any firmware updates which are available from the FUS

3.2 Abbreviations

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

ABNF	Augmented Backus-Naur Form
ACS	Auto-Configuration Server
AES	Advanced Encryption Standard
ALC	Asynchronous Layered Coding
ASM	Any Source Multicast
B2B	Business To Business
BBF	Broadband Forum
CDS	Content Download System
CE	Consumer Electronics
CPE	Customer Premises Equipment
CWMP	CPE Wan Management Protocol
DNG	Delivery Network Gateway
DSM-CC	Digital Storage Media - Command & Control
DVB	Digital Video Broadcasting
DVBSTP	DVB SD&S Transport Protocol
FEC	Forward Error Correction
FLUTE	File deLivery over Unidirectional Transport

FTP	File Transfer Protocol
FUS	Firmware Update System
FUSS	FUS stub file
HN	Home Network
HNED	Home Network End Device
HTTP	Hyper Text Transfer Protocol
HTTPS	Secure HyperText Transport Protocol
IANA	Internet Assigned Numbers Authority
ID	IDentifier
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IP	Internet Protocol
IPI	Internet Protocol Infrastructure
IPv4	Internet Protocol version 4
ISO	International Organization for Standardization
LCT	Layered Coding Transport
MAC	Media Access Control, depending on context
MIME	Multipurpose Internet Mail Extension
MPEG	Moving Pictures Expert Group
MPTS	Multiple Program Transport Stream
MTU	Maximum Transmission Unit
NTP	Network Time Protocol
OSS	Operations Support System
QRC	Query/response channel
RFC	Request For Comments
RMS	Remote Management System
RPC	Remote Procedure Call
SAP	Service Announcement Protocol
SD&S	Service Discovery and Selection
SDP	Service Description Protocol
SOAP	Simple Object Access Protocol
SP	Service Provider
SSM	Source Specific Multicast
SSU	System Software Update
STB	Set Top Box
TCP	Transmission Control Protocol
TFTP	Trivial File Transfer Protocol
TLS	Transaction Layer Security
TSI	Transport Session Idendifier
TSI	Transport Session Identifier
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WWW	World Wide Web
XML	eXtensible Markup Language

3.3 Notations

3.3.1 Augmented Backus-Naur Form (ABNF)

The present document uses the Augmented Backus-Naur Form (ABNF) conform to [28], for syntax specification.

4 Reference Model

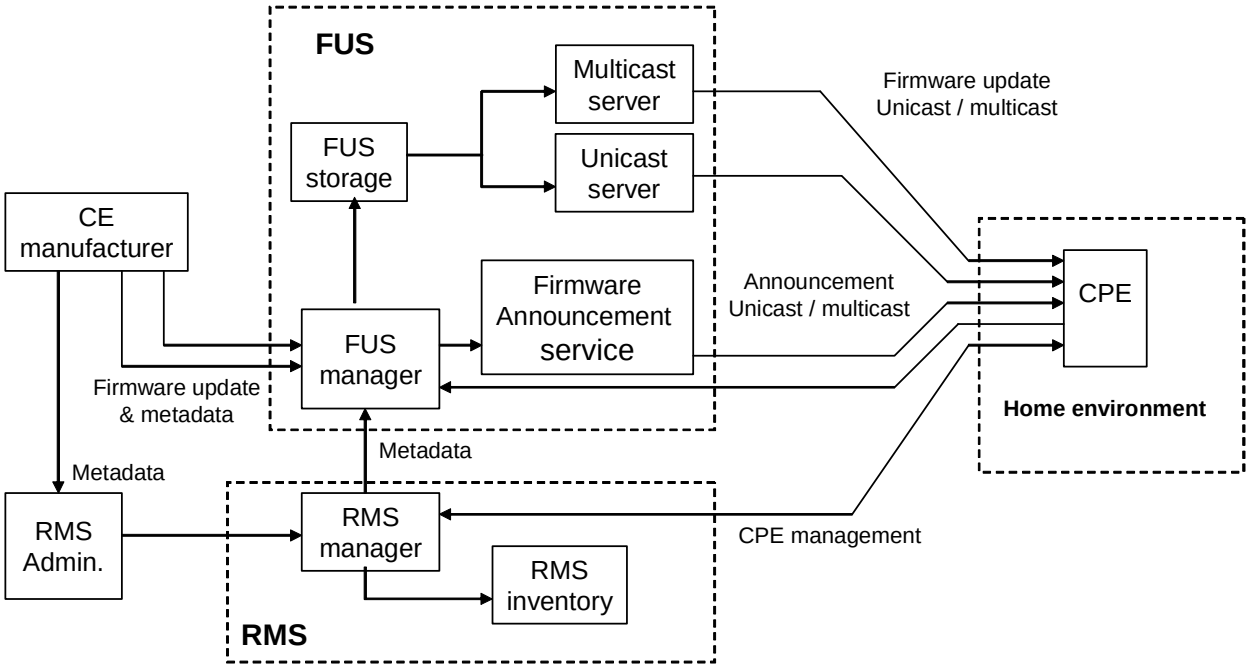

Figure 1: Overview of RMS-FUS environment architecture

The functional model as shown in figure 1 enables a range of methods for CPEs to be remotely managed and provided with firmware updates. These methods are:

- RMS-only for managed devices where there is no support for firmware updates;
- RMS-FUS for managed devices where firmware update support is available;
- FUS-only for delivering firmware updates to unmanaged CPEs.

All of these methods may be applied at and after boot time for a CPE.

This range of capabilities is made possible by separating the logical entities associated with the remote management system (RMS) from those required for the firmware update system (FUS) entity, and defining each of them independently of the service package aggregation, network operation and delivery. The minimum standardisation is specified to maintain inter-operability.

The actual level of functionality offered by the various solutions will depend on the requirements of the RMS administrator and the capabilities of the associated RMS and FUS.

4.1 Modes of operation

The significant difference between the RMS-FUS methods is summarised by describing the abilities of the RMS and FUS entities:

- An RMS is a management entity that interacts with individual CPEs:
  - If the use case requires a network entity to have knowledge of and in defined cases to be able to exert control over individual CPEs an RMS is needed and all management communication between RMS and CPE will be via the CPE management interface.
  - If the firmware update requirement is for a more complex update sequence which cannot be managed by a CPE behavioural model based on the description metadata it must be done through an RMS.

- The RMS will use the CPE management interface but may also use the multicast announcement service and unicast query/response channel in managing the populations of CPEs.
- Either the RMS or CPE may initiate the firmware update based on a communication over the CPE management channel, or the CPE may query the FUS over the query/response channel.
- The DVB RMS does not have the capability to deliver the actual firmware update file.

- A FUS is a firmware update entity that is not aware of individual CPEs:
  - Firmware updates which can be delivered based on the information carried in the notification service or identified by a simple query can be supported by an FUS-only system without an RMS.
  - In general terms responses by the FUS to CPE queries will be based on comparisons done between CPE status information provided in the query and the metadata provide by the CE manufacturer with firmware update files.
  - Communications between the CPE and FUS will take place over the combination of the multicast announcement service and the unicast query/response channel.
  - The CPE may send a query to the FUS over the query/response channel to check the availability of a firmware download, the FUS cannot initiate the message sequence with a CPE.
  - The DVB FUS has the capability to deliver the actual firmware update file in either a multicast or unicast way.

4.1.1 RMS-only mode

This mode will be used to allow a service provider or other agency to manage a population of CPEs on an individual basis because it is assumed the managing authority (the RMS administrator) has knowledge of that population of CPEs. Figure 2 shows an overview of the logical RMS-only architecture.

The ability to maintain the firmware in terms of updates using a mechanism specified by DVB is not supported in this mode.

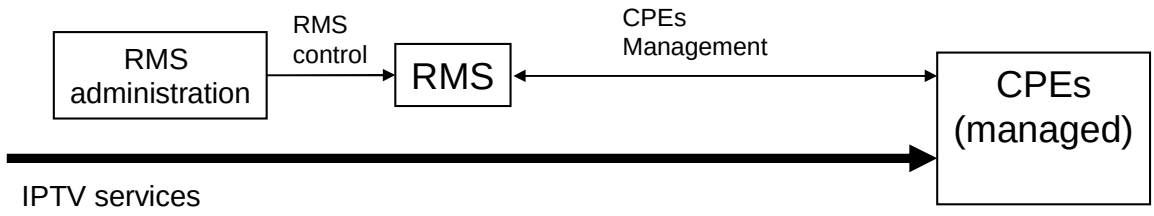

Figure 2: Overview of system architecture of RMS only environment

4.1.2 RMS-FUS mode

The FUS may be added to manage the ongoing firmware status of a population of CPEs managed by that RMS. This is then the RMS-FUS mode, although the scope of the services offered by any RMS is not specified in the present document. Figure 3 shows an overview of the logical RMS-FUS architecture, the "CPEs (unmanaged)" block shown in the diagram is included to illustrate that a single FUS might supply firmware updates to both managed and unmanaged CPEs simultaneously.

The RMS assumes control of the functionality of the FUS relevant to managing that population of CPEs under the control of that RMS. It shall be able to initiate download operations and request responses from individual CPEs using the CPE management interface.

Firmware updates initiated by an RMS may be carried in either a unicast or a multicast way over the network.

The RMS may manage the operation of reconfiguring the firmware download arrangement for specific managed CPEs which are having problems using the basic delivery mechanism.

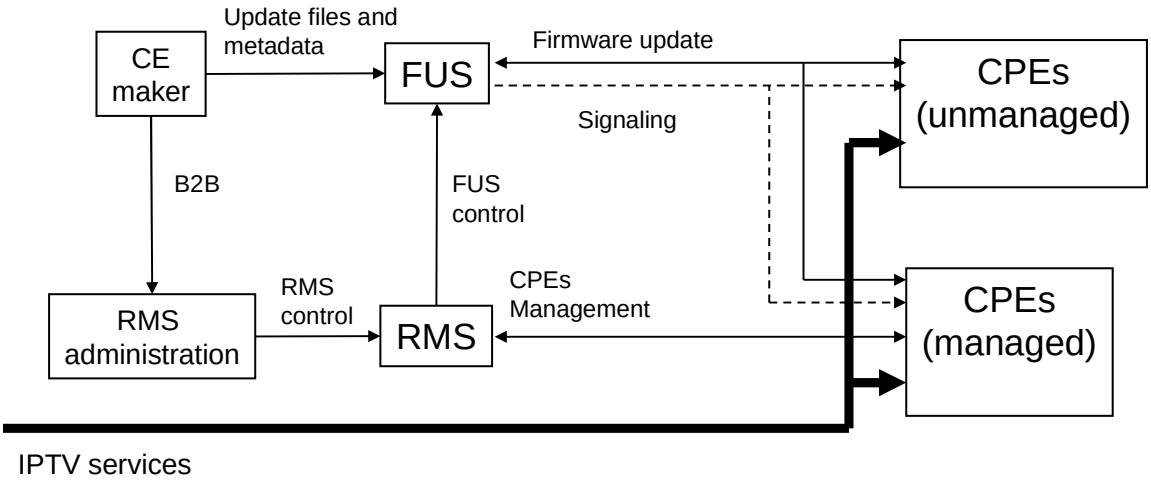

Figure 3: Overview of system architecture of RMS-FUS environment

In addition to using the RMS for maintaining the current status of the firmware version in the CPE, the CPE management channel makes it possible for the RMS to manage a complete population of CPEs for a range of aspects including complex update sequences.

4.1.3 FUS-only mode

The FUS-only mode will be used to allow unmanaged CPEs to get firmware updates over the network. Figure 4 shows an overview of the logical FUS-only architecture.

In this mode the CPEs will either receive notifications of new firmware versions available from the FUS by monitoring the multicast announcement service or directly query the FUS to check whether suitable new firmware versions are available using the FUS addresses provided at boot time in the FUS Stub file (FUSS).

There is no specified mode for the FUS to initiate a message to any CPE without an initial query from that CPE, and it should not be assumed that an FUS will contain a CPE inventory in any form. However, decisions based on simple comparisons using the metadata provided by the CE manufacturer with the firmware update file should be possible in this FUS-only mode.

Firmware updates for FUS-only mode may be carried in either a multicast or unicast manner. Mechanisms to configure alternative firmware update delivery arrangements may be possible to help to mitigate in the case of CPEs which repeatedly fail to download an update file correctly. These methods shall depend on a combination of programmed CPE behaviour and signalling in the metadata received by the CPE.

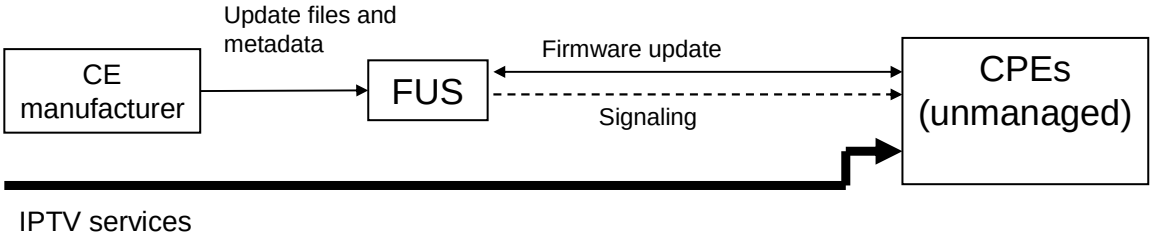

Figure 4: Overview of system architecture of FUS-only environment

4.2 Type of firmware update files

Depending on the structure of the firmware stack in the CPE and the policy of the CE manufacturer and RMS (for a managed CPE), a single firmware update image file may be one of several options:

- Full firmware stack;
- Application;

- Middleware;
- Middleware module;
- Management system;
- Boot block;
- Configuration.

Note that this list is not exhaustive, and although information about the type of update is contained in the XML firmware update description, the information needed (image type, installation address, etc.) for the CPE to make use of the image in the CPE is considered to be manufacturer and instance specific and shall be carried within the file.

4.3 The role of the entities in the RMS-FUS architecture

All the entities identified within the FUS and RMS sub-systems are logical rather than physical in nature. No specific physical device is implied.

The architecture supports firmware maintenance for any type of CPE in a managed or unmanaged environment provided they support the present document.

The functionality of the RMS and FUS sub-systems is specified in clause 5, and the details of the interfaces in clause 6.

4.3.1 CPEs

The CPE is the device in the home environment which is managed by the RMS and for which firmware updates are provided by the FUS.

CPEs which are intended to be provided with firmware updates over the network in a DVB specified manner (either managed or unmanaged) should support the appropriate delivery protocols and signalling methods as specified in clause 6 of the present document.

CPEs managed by a remote management system for use in a managed network shall support the CPE management interface described in the present document. This may include the process of firmware update for the CPE in which case the FUS interfaces as defined in the present document.

4.3.2 RMS Administrator

Information about a managed population of CPEs will be held by an RMS administrator based on product installation information, service changes, etc.

This RMS Administration sub-system will control the activities of the RMS. There may also be a business communication between the RMS Administration and the CE manufacturer.

Specification of both of these interfaces is outside of the scope of the present document.

4.3.3 RMS

The RMS sub-system, for which the functionality is described in clause 5.2 is responsible for managing, enforcing, and modifying the behaviour of the population of CPEs under its control. The purpose of an RMS is to:

- Manage individual CPEs in terms of:
  - Provisioning:
    - Configuration.
  - Assurance:
    - Diagnostics;
    - Troubleshooting;

  - CPE monitoring;
  - Fault management.

- Coordinate the operation of the FUSs over which it has some control for the population of CPEs for which it has control.
- Pass on CPE status information to additional agencies and the OSS of the same agency.

An RMS deals with individual CPEs and, as such, may require access to an inventory of managed CPEs; information to populate the inventory will be provided by the RMS administration.

The following types of interactions can be realized on the CPE management interface:

- RMS-initiated command/request/response.
- CPE-initiated notification, e.g. on boot or on completion of a download.

At any instant an individual CPE may only be managed by one RMS, using the CPE management interface, and all management functions should be carried out through this process.

4.3.4 CE manufacturer

The firmware updates will be made available by the manufacturers of the CPEs - the CE manufacturers. The CE manufacturers must also provide the descriptive metadata associated with the firmware update files. The logical interfaces are specified in the present document, but the method of delivery used by the CE manufacturer for the metadata and firmware file is not specified in the present document.

In an actual implementation the image files and metadata may be supplied to the FUS and RMS by either the CE Manufacturer or an intermediate agent.

4.3.5 FUS

A FUS deals with classes / collections of CPEs without any knowledge about individual CPEs and to be compliant with the present document is a content server which is capable of:

- Receiving metadata associated with new firmware update files from the CE manufacturer.
- Downloading firmware updates from the CE manufacturer based information contained in the metadata provided.
- Storing firmware update files from the CE manufacturer.
- Creation and maintenance of the downstream announcement signalling describing available firmware updates based on the information supplied in the metadata.
- Optionally being managed by one or more RMSs, as well as operating in an FUS-only mode for different CPEs.
- Responding to queries from a CPE containing manufacturer, model, current version, serial number, etc. in order to get the most appropriate firmware update for that CPE if one is available.
- Managing the configuration of delivery of the firmware files to the network so that the CPEs can download them in an appropriate way.
- Distributing specified firmware update files using multicast or unicast delivery as specified in the present document.

The metadata provided by the CE manufacturer may contain some or all of the element groups below, and without this information the performance of the announcement and delivery cannot be optimised:

- Mode - RMS-FUS or FUS-only.
- Entity description characteristics of the CE manufacturer, RMS and FUS.

- Targeting - characteristics of the CPEs for which the update is suitable.

Note that except for the information contained in this associated metadata it shall not be assumed that the FUS has any knowledge about individual CPEs or CE manufacturers to use to create entries for the signalling or to reply to queries. In the absence of sufficient metadata the default behaviour is not specified.

The control and triggering of the playout of the update file streams over multicast or unicast will be controlled by the FUS using the information carried in the associated metadata from the RMS in the RMS-FUS environment, although in the absence of an RMS for part of the population of CPEs the metadata supplied directly by the CE manufacturer could be used.

The method of implementation of the FUS is out of scope for DVB but in the present document DVB specifies the required capabilities of some of the interfaces to other sub-systems in order to maintain inter-operability.

The functionality of the blocks within this sub-system is described in clause 5.

4.4 Control priorities of RMS vs. FUS-only

An RMS-FUS environment may include multiple RMSs controlling different populations of CPEs, as well as some unmanaged CPEs which shall be able to operate using the signalling and messaging within the capability of the FUS.

If an RMS is present which manages all or part of the population of CPEs supported by the FUS, then that RMS shall take precedence over any RMS or FUS interactions with the CPEs in the population which are managed by that RMS.

Other CPEs may use either an alternative RMS to control the CPEs under their management, or work in an unmanaged way with no reference to any RMS through the same FUS.

As part of the business-to-business relationship, the RMS may be required to share CPE status data with other agencies, e.g. CE manufacturers. Note however that an individual CPE may only be managed by one RMS at a time.

4.5 CPE Bootstrap process

4.5.1 FUSS Process (Managed and unmanaged CPEs)

Bootstrap is the method by which the CPE identifies the initial entry point into the FUS server using information made available through the FUS stub file (FUSS) and is described in clause 9 of [1].

For any CPE the URI or address provided in the FUS stub (FUSS) will lead to the entry point from where a firmware update may be identified (if available), located, downloaded and installed.

That URI or address shall provide one or more of the following:

- The multicast address which will provide the location of one of the following:
  - The announcement service from which the identity and location of the download image file can be found.
  - The firmware update image file.
- The unicast address which will provide the location of one of the following:
  - The firmware update image file.
  - The query/response channel.

NOTE: The use of a multicast address does not preclude the use of the unicast query/response channel since the URI of the QRC may also be obtained from the metadata.

4.5.2 Process after FUSS (Managed CPEs)

In case of managed devices the RMS has to be contacted by the CPE during the boot process. Although DVB is in principle neutral regarding the remote management protocol, CWMP (BBF TR-069 protocol) is considered hereafter since this is the recommended approach. However, other management protocols would be possible.

According to the TR-069 framework the RMS may configure the CPE with either the URL of the FUS multicast announcement channel (.DownloadAvailability.Announcement) or the URL of the QRC (.DownloadAvailability.Query), or both. In the DVB framework the RMS configured addresses will overwrite the address received with the FUSS, i.e. the RMS provided addresses may be cached for future use.

The TR-069 framework also allows configuring the CPE policy for notification of AUTONOMOUS TRANSFER COMPLETE events. This policy determines the conditions under which the CPE notifies the RMS of the completion of file transfers that were not specifically requested by the RMS.

According to TR-069 the CPE must contact the RMS during the boot process using an "Inform" RPC to report about the boot event. If in addition the CPE has downloaded an upgraded firmware file e.g. as a consequence of the examination of the FUSS, then also this event needs to be reported e.g. as follows:

- "1 BOOT".
- "10 AUTONOMOUS TRANSFER COMPLETE".

In the same session the CPE will also call the AutonomousTransferComplete RPC to inform the RMS of the completion (either successful or unsuccessful) of a file transfer that was not specifically requested by the RMS.

The event " AUTONOMOUS TRANSFER COMPLETE " and the AutonomousTransferComplete RPC have been newly defined in CWMP 1.1 (TR 069 Amendment 2) in order to accommodate specific DVB needs.

The RMS may configure the CPE with either the URL of the FUS multicast announcement channel (.DownloadAvailability.Announcement) or the URL of the QRC (.DownloadAvailability.Query), or both. In those cases the RMS configured addresses will overwrite the address received with the FUSS, i.e. the RMS provided addresses may be cached for future use.

The RMS may also configure the CPE policy for notification of AUTONOMOUS TRANSFER COMPLETE events. This policy determines the conditions under which the CPE notifies the RMS of the completion of file transfers that were not specifically requested by the RMS.

4.6 Running state behaviour of CPEs

4.6.1 Unmanaged CPEs

After the boot process has completed and the CPEs are working in the normal state the methods described in the present document allow the CPEs to continue to monitor for the availability of firmware updates. The present document does not define the operational behaviour of the CPEs except in terms of the interfaces with the RMS-FUS services.

Options which may be used by unmanaged CPEs for monitoring for firmware updates are:

- To monitor the multicast announcement service delivered using either DVBSTP/UDP or SDP/SAP/UDP, this will return location information for a firmware update if one is available.
- To use a SOAP/HTTP query to the FUS over the QRC containing identification information about the CPE, the response will return location information for a firmware update if one is available.

Information may be provided about an optional reporting channel ("reporting_orders") independent of the FUS but by which a CE Manufacturer may get feedback about the installation of an update either directly or through a third party.

4.6.2 Managed CPEs

Managed CPEs may use the CPE management channel based on methods defined in the appropriate RMS documents, also in addition to the options which are described above for the unmanaged CPE. It is assumed that the CPE will report update success/failure back to the RMS controlling it using one of the mechanisms specified by the RMS.

5 The RMS-FUS sub-systems

The reference architecture in figure 5 extends the overall schematic of the RMS-FUS architectures (figures 2, 3 and 4) to show additional detail of the functional blocks and interfaces for the FUS and RMS sub-systems. In the RMS-only and FUS-only options the appropriate interfaces will show the same capabilities and behaviour.

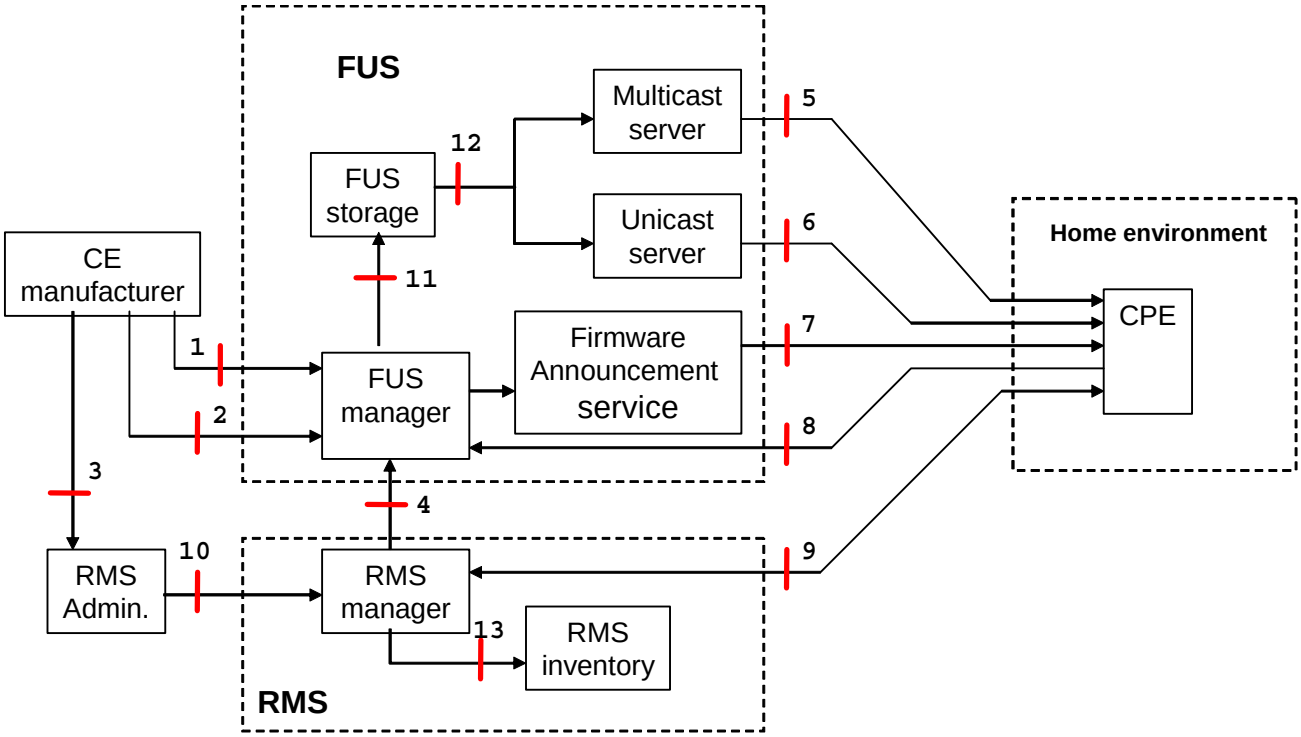

Figure 5: Architecture showing system interfaces

The FUS and RMS subsystems and interfaces, as well as some further details of both internal and external interfaces have been exposed to show logical connections (e.g. interfaces 1 and 2 are shown as separate).

Clause 5 describes the functionality and interfaces of the sub-systems and clause 6 describes the technical requirements.

The arrow direction indicates the device status using the client/server model - the arrows go from the client to the server. Lines with arrows at both ends indicate that both entities can act either as client or as server over these interfaces.

5.1 Interfaces overview

Table 1 following contains an introduction to the function of each interface and the aspects which are standardised as either mandatory or optional.

Table 1: Overview of interface functions

Interface number	Interface name	Description	Scope in terms of the present document
1	Firmware package	This carries the files containing the firmware from the CE manufacturer to the FUS will be carried on this interface if the standardised mechanism is used.	Out of scope.
2	Metadata	Metadata provided by the CE manufacturer to describe the properties of the firmware update package for the RMS and FUS.	Schema Definition (XML) standardised, use across this interface is recommended.
3	CE manufacturer - RMS administrator	B2B relationship between CE manufacturer and RMS administration.	Schema Definition (XML) standardised, use across this interface is recommended.
4	RMS-FUS control interface	Metadata passed from FUS to RMS and RMS to FUS to manage the download behaviour.	Schema Definition (XML) standardised, use across this interface is recommended.
5	Multicast delivery	Multicast delivery as a service over the network to the population of CPEs.	Transport protocol options standardised. Authentication of source recommended, method not specified. Payload protection is not standardised. Refer to clause 6.5 for detail.
6	Unicast delivery	Unicast delivery as a service over the network to the population of CPEs.	Transport protocol options standardised. Authentication of source and destination recommended, method not specified. Payload protection is not standardised. Refer to clause 6.6 for detail.
7	Firmware announcement interface	This service carries notification information about firmware updates which are available over the network.	Schema Definition (XML) standardised Transport protocol options standardised Authentication of source recommended, method not specified Payload protection is not standardised Refer to clause 6.7 for detail.
8	Query Response Channel (QRC)	In the FUS-only model this enables the CPE to query the FUS to find out if firmware updates are available for CPE.	Transport protocol and RPC arguments standardised. Authentication of source and destination recommended, method not specified. Refer to clause 6.8 for detail.
9	CPE management	This interface may be compliant with BBF TR-069 with DVB specified extensions (recommended solution) or other functionally equivalent protocols such as Cable Labs PACM with extensions (not specified in the present document).	Profile and extensions to TR-069 [6] specified by DVB in cooperation with BBF. Refer to clause 6.9 for detail.
10	RMS administration	B2B interface between RMS administrator and RMS.	Out of scope.
11	FUS storage	Internal to FUS.	Out of scope.
12	FUS processing	Internal to FUS.	Out of scope.
13	RMS inventory	Internal to RMS.	Out of scope.

5.2 The Remote Management System (RMS)

This clause describes the functionality of the logical modules in the RMS sub-system in terms of specifying their interaction with the other modules and sub-systems within a managed environment.

The total functional capability of a specific RMS for a specific RMS-FUS environment is not specified in the present document.

5.2.1 Communication with RMS administration

Communication between RMS and RMS administration is done through interface 10. The requirements for this is out of scope of this DVB document in terms of both transport protocol and messaging format.

5.2.2 Management of the inventory

The RMS inventory (interface 13) is internal to the RMS; therefore all aspects of this functionality and the communication over interface 13 is out of scope for the present document.

5.2.3 CPE management interface

In a managed network it is recommended that the RMS shall manage the CPEs by using the methods specified in TR 069 [6] and associated documents, including extensions created within the development of the present document and profiled in annex A. The CPE management interface is shown as interface 9 in figure 5.

Other existing protocols such as PACM (using SNMP), for which the specification is outside the scope of the present document, may also be used where it is understood by appropriate.

When Broadband Forum methods are used the CPE management protocol (TR 069 CWMP) delivers the following functionalities, other management protocols like PACM could provide equivalent functionalities:

1) RMS and CPE may both initiate or request to initiate the communication for management purposes;

2) RMS may read the complete configuration of each managed CPE;

3) RMS may change the complete configuration of each managed CPE;

4) CPE may send status report to the RMS, e.g. at each boot, at each specified event;

5) RMS may request the execution of diagnostics tests on the CPE and collect the result;

6) RMS may invoke operational commands on the CPE, e.g. Reboot, FactoryReset;

7) RMS may configure on the CPE the requested behaviour in terms of active/passive/scheduled notification of events/alarms, e.g. for fault management, performance management, SLA management, statistics collection;

8) CPE may autonomously send events/alarms to the RMS, in compliance with the configured behaviour;

9) RMS may command the CPE to download and apply a file, e.g. a configuration file or CPE Firmware. Files can be downloaded by means of unicast (e.g. HTTP) as well as multicast (e.g. FLUTE) protocols. After download the file is applied as intended by the CPE. This may cause the CPE to autonomously reboot, e.g. in case of Firmware update;

10) RMS may command the CPE to upload a file, by means of a unicast protocol (e.g. HTTP);

11) CPE may inform the RMS of the success/failure of any commanded file transfer (download/upload);

12) CPE may autonomously start a firmware update process via a explicit query to the RMS.

NOTE: CPE may also autonomously carry out a firmware update as a result of the multicast announcement or a query over the QRC.

13) CPE may inform the RMS of the successful/unsuccessful completion of the autonomous firmware update process;

14) RMS may implement recovery mechanisms to respond to download failures or configuration problems.

The method by which these functionalities may be implemented via TR-069 / CWMP is described in the relevant document [6].

5.2.4 Processing of metadata and communication with the FUS (interface 4)

The RMS must receive metadata describing all relevant aspects of the firmware update files from the CE manufacturer or other agent who produces and makes them available via the RMS administrator (interfaces 3 and 10). That information shall be used by the RMS to modify and produce a metadata subset of the schema described in annex B (delivery metadata), describing how the FUS shall advertise and deliver the firmware updates referenced by that metadata.

The FUS shall deliver the update file as described in the metadata. In the presence of an RMS, the FUS will be expected to follow the instructions provided in the delivery metadata or by other means.

This interface is shown as interface 4 in figure 5 and described in clause 6.4 in more detail.

5.3 The Firmware Update System (FUS)

This clause describes and specifies the functionality of the logical modules in the FUS sub-system as shown in figure 5. The diagram is conceptual and is not intended to describe any actual implementation.

5.3.1 FUS manager and storage modules

The function of the FUS manager is to coordinate the download of the firmware update files from the CE manufacturer over interface 1 in accordance with the associated metadata delivered from the CE manufacturer to the FUS manager over interface 2, arrange the process of storage, and supply the appropriate files to the multicast and unicast servers when required with the appropriate IPTV configuration.

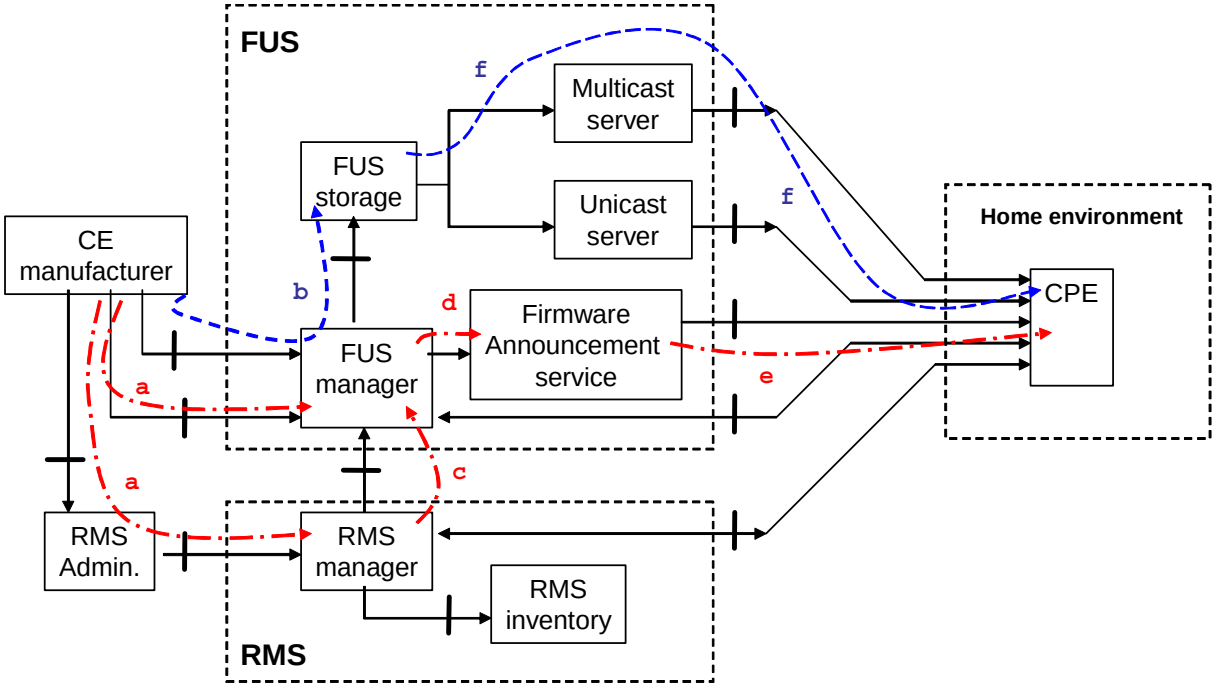

Figure 6: Metadata and firmware update flow sequence for firmware update acquisition and multicast streaming

Figure 6 shows the message and firmware update file interchange starting with the metadata sent from the CE manufacturer indicating that an update is available, through to the FUS creating the announcement message to the CPEs and the multicast delivery service being available. The control of the FUS may either be by the RMS or based on that of a simple server using the available metadata.

The actions identified by the numbers (in sequential order) in figure 6 are described in table 2. The data flows in red represent metadata and those in blue the firmware update, and also show an example of a the resulting multicast delivery.

Table 2: Description of actions for figure 6

Action identifier	Description
a	CE manufacturer sends message including metadata to FUS and to RMS describing new firmware update available.
b	FUS may download file, note all the following actions assume that this is done. The firmware file will be prepared for the servers.
c	RMS sends delivery metadata back to FUS to manage firmware updates to managed CPEs.
d	FUS creates entry in the announcement interface describing firmware update.
e	Firmware announcement is streamed (multicast) to population of CPEs.
f	FUS starts multicast download service, diagram includes connection of CPE to multicast delivery.

A DVB compliant FUS manager shall be capable of parsing the metadata from the CE manufacturer to manage the processing and delivery of the firmware update file made available. In the case where there is an RMS for a given population of CPEs and where that RMS will make use of a firmware update file the delivery metadata shall be used to create entries in an announcement service. In the FUS-only case the metadata from the CE manufacturer shall be used to manage the creation of notification channel entries and the firmware delivery service.

The FUS manager must be capable of the processing, fragmentizing, etc. the file in preparation for delivery to one or more CPEs in a DVB compliant way.

The FUS manager functionality shall optionally include the FUS part of the query/response channel (interface 8), although the query/response channel is required if the FUS is to support unmanaged CPEs.

The present document defines various metadata exchanged between various entities, e.g. RMS, FUS, CPE. The exact metadata is a function of which entities are present and message exchange profile. The resulting metadata set for any specific implementation will depend on the entities present and the interfaces implemented.

The FUS XML metadata schema is defined in annex B, and as a general rule:

- CE manufacturer is responsible to compile the CEManufacturerInfo, TargetDeviceInfo and FirmwareUpgradeInfo elements.
- FUS is responsible to compile the FUSInfo element, and may update the FirmwareUpgradeInfo element.
- RMS is responsible to compile the RMSInfo element, and may also update the FirmwareUpgradeInfo element. In the RMS managed case any changes carried out by the RMS will take precedence over those carried out by the FUS.

5.3.2 Signalling and messaging to enable CPEs to locate updates

The FUS must provide the signalling and messaging to enable the CPE to locate the appropriate update files and the associated delivery information. This information shall be made available in two forms:

- Announcement service: one or more multicast delivery services carrying information about all firmware update files available.
- Query/response channel (QRC): a message channel with message sequences being initiated by the CPE. This will offer limited functionality but in the FUS-only mode will allow CPEs to find out about available firmware updates from the FUS by using a query.

5.3.2.1 Multicast announcement service

The FUS may distribute firmware announcement information via one or more multicast channels. The HNEDs may join these channels based on information provided for example by the FUS stub file. The announcement data shall be created and maintained by the FUS based on available FUS metadata.

The firmware announcement service is the downstream only service by which the FUS will announce information about available firmware updates, it must be supported by all DVB compliant FUS implementations and CPEs. It shall be created and maintained by the FUS based on available metadata.

The announcement service is shown in figure 5 as interface 7 and described in detail in clause 6.7.

5.3.2.2 Unicast query/response channel (QRC)

The query/response channel is shown as interface 8 in figure 5 and described in detail in clause 6.8. It is able to carry queries originated by a CPE to the FUS to identify whether a firmware update is available based on the information supplied in the query and the metadata provided by the CE manufacturer.

No capability for the FUS to initiate a message interchange with the CPE is included in this interface.

5.3.3 Firmware update delivery

The FUS shall be able to deliver the payload in both multicast and unicast modes to the CPEs in the home environments. The description of the technical requirements in terms of this DVB document for these delivery mechanisms is given in clauses 6.5 and 6.6.

In a managed environment the delivery services will be configured as a result of a command from the RMS for a download service to managed CPEs, or in an unmanaged environment in accordance with a description given in the update announcement interface for that service or as a result of a query from a CPE to the FUS.

Firmware updates to populations of managed CPEs will be managed by an RMS.

5.3.3.1 Delivery failure in a managed environment

If a managed CPE is not capable of successfully completing a firmware file download an appropriate message returned on the CPE management interface may be used to inform the RMS of the problem, and it is the responsibility of the RMS to take appropriate action in these circumstances. Based on this message the RMS may instruct the FUS to configure alternative download services, e.g. slower multicast or targeted unicast services, for those CPEs, and supply the appropriate locators for those download services.

The control messages to manage the process between the RMS and the CPE will be carried over the CPE management interface and the capability of the recovery options is out of the scope of the present document.

5.3.3.2 Delivery failure in an unmanaged environment

The FUS may provide several locators with usage preferences either in the response to a query from a CPE or in the multicast announcement service. The CPE may make use of those alternatives if the first preference fails.

Either multicast or unicast service options may be offered by the FUS. The operation of this mode depends on a combination of programmed CPE behaviour and information supplied in the metadata by the FUS.

5.4 Secure Message Exchange and Secure Network Time

The clause describes the interoperation requirements for Secure Message Exchange and Secure Network Time.

If authentication is required to set up a connection with the file server for either unicast or multicast, the CPE must be provided with correct credentials. This provisioning may be done by one of several means including those listed below:

- Factory default configuration.
- Local configuration (e.g. via a GUI or a Smart Card).

- Configuration using the RMS interface for managed CPEs.

The way the CPE is configured and shares the credentials with the FUS and the RMS is out of the scope of this clause.

5.4.1 Secure Message Exchange

The Secure Message Exchange adopts the Transport Layer Security (TLS) specifications [16] and [15].

The protocols require the selection of ciphersuites and those selected are listed in table 3.

The applicable references for the RSA_WITH_AES_128_CBC_SHA and RSA_WITH_AES_256_CBC_SHA ciphersuites (see table 3) [13] and [16].

The secure message exchange requirements are a function of the specific interface. The recommendation of the present document is that if the control interfaces, that is the Query Response Service (Interface 8) and the Remote Management Service (Interface 9) elect to support secure message exchange, the interfaces adopt the ciphersuites below.

NOTE: The notation "RSA_WITH"_ in the table implies that both TLS and SSL dialects are applicable. The RSA_WITH_3DES_EDE_CBC_SHA ciphersuite, for example, means both the TLS_ RSA_WITH_3DES_EDE_CBC_SHA and the SSL_ RSA_WITH_3DES_EDE_CBC_SHA ciphersuites are applicable.

Table 3: Required compliance with ciphersuites for QRC and RMS

Ciphersuite	Query Response Service (Interface 8)	Remote Management Service (Interface 9)
RSA_WITH_RC4_128_SHA	C(M)	C(M)
RSA_WITH_3DES_EDE_CBC_SHA	C(M)	C(M)
RSA_WITH_AES_128_CBC_SHA	C(S)	C(S)
RSA_WITH_AES_256_CBC_SHA	C(S)	C(S)

The notation is:

C(M) = If the interface supports TLS:TCP:IP, support this ciphersuite is mandatory.

C(S) = If the interface supports TLS:TCP:IP, the interface should support this ciphersuite.

The inclusion of the RSA_WITH_RC4_128_SHA and the RSA_WITH_3DES_EDE_CBC_SHA ciphersuite aligns the present document with TR-069 [6] which requires these for TLS:TCP:IP stacks. See clause 6.9 for further details. The same ciphersuites may be used for the query/response channel on interface 8, see clause 6.8.

The interoperation requirements for Secure Message Exchange are a function of the specific interface and each interface is considered in the security clauses of those interface descriptions based on the protocols used and the specifications to which they are compliant, but using methods described in this clause.

5.4.2 Secure Network Time

The CPE and FUS should support secure network time using Network TimeProtocol v3. Note that the field that encodes the signature algorithm is carried in the NTP packet structure.

The FUS and CPE shall synchronize against the network time through the NTPS:TCP:IP stack, [17]. The checksum of the protocol presumes the Message Digest 5 algorithm, based on [18].

6 The interfaces

Thirteen interfaces are defined (shown in figure 5) as being relevant to the model, although some of these are not standardised by DVB. These are referred to by numbering (1 to 13) in figure 5. The specification of some of these interfaces is not in scope of the present document.

Some interfaces, such as interface 5 for multicast and 6 for unicast, are mandatory for both the RMS-FUS and FUS-only modes of operation, and other shall be used to manage and coordinate the firmware download operations. There are also different requirements and needs among the server side portion of the architecture (involving both the FUS and the RMS entities) and the client side of the architecture (involving CPEs).

The following tables identify minimum requirements in terms of the implementation of some interfaces and assure interoperability among server side and client side accordingly with mandatory requirements. They are based on the following assumptions:

- Multicast firmware update on interface 5 is required to be announced by interfaces 7, 8 or 9.
- Unicast firmware update on interface 6 is required to be announced by interfaces 7, 8 or 9.

Each row of the table shows a possible combination of interfaces. Additional features over and above those combinations required for conformance optionally add some functionality to the RMS-FUS system.

Table 4: Interface conformance - server side

Server side requirements							
	Interfaces					Conformance to DVB RMS-FUS specification	Comments
	5	6	7	8	9		
RMS-FUS	X	X		X	X	Not conformant	Interface 7 is required for the multicast announcement.
	X	X	X	X	X	Optional	Full implementation is optional.
	X	X	X		X	Required	Interface 9 is required to be able to replace all functionalities of interface 8.
FUS-only	X	X		X		Not conformant	Interface 7 is required for the multicast announcement.
	X	X	X			Not conformant	Unicast without interface 8 is not able to offer complete functionality.
	X	X	X	X		Required	Minimum set of required interfaces.

The rows in the following CPEs requirements table show which combinations of interfaces are granted to interoperate with the server side of the architecture. It is the responsibility of the CE manufacturer to decide which combination is to be implemented. For example, if a CE manufacturer wants to implement a CPE which is not capable of unicast download, the first row of unmanaged section or the first row of managed section shall be considered, otherwise the unicast download functionality shall be also offered. The combinations are separated into two groups covering managed CPEs and unmanaged CPEs.

Table 5: Interface conformance - CPE side

CPEs requirements							
	Interfaces						
	5	6	7	8	9		
Managed	X		X		X	Optional	Only one of these interfaces combinations must be supported by the managed CPE.
	X	X	X	X	X	Optional	
		X		X	X	Optional	
		X			X	Optional	
	X	X	X		X	Optional	
Unmanaged	X		X			Optional	Only one of these interfaces combinations must be supported by the unmanaged CPE.
		X		X		Optional	
	X	X	X	X		Optional	

6.1 Interface 1 - Firmware Update file from CE manufacturer to FUS manager

This is a business to business (B2B) interface carrying the firmware update files from the CE manufacturer repository to the FUS. The files will be delivered as demanded by the FUS.

6.1.1 Payload coding and format

The CE manufacturer may choose to protect the firmware update file using a mechanism such as encoding or encryption in a way which is only known by the CPEs it is intended for. There shall be no requirement for the FUS to be able to access the information in the update file.

6.2 Interface 2 - Associated metadata from CE manufacturer to FUS manager

The CE manufacturer may use interface 2 to provide metadata with the firmware update to the FUS. That metadata should carry information describing the firmware update and how it should be delivered to the CPEs. The metadata flow is as shown in figure 6. The XML metadata schema is defined in annex B.

The CE manufacturer may use interface 2 to provide metadata with the firmware update to the FUS. That metadata should carry information describing firmware update and how it should be delivered to the CPEs. The metadata flow is as shown in figure 6 and is based on the schema in annex B.

The delivery of the metadata is out of scope of the present document.

6.3 Interface 3 - CE manufacturer to RMS

This is a business to business (B2B) interface. To make it possible to manage the population of CPEs under its control the RMS must have knowledge about the firmware update files which are available. This information delivery is not appropriate if an FUS-only mode is used, and the information provided shall be identical to that provided by the CE manufacturer to the FUS.

This metadata is used to manage one or more of FUS sub-systems to provide the firmware updates for managed and unmanaged CPEs in the home environment.

The metadata flow is as shown in figure 6, and the schema used shall be based on the complete schema included in annex B.

The delivery of the metadata is out of scope of the present document.

6.4 Interface 4 - RMS-FUS control interface

As described in clause 6.3 the RMS will receive the same metadata from the CE manufacturer as supplied to the FUS. This will allow the RMS to control the managed CPEs for which it is responsible in terms of instructions to the FUS (Delivery Metadata) controlling the announcement service for that managed population and the configuration of the firmware delivery services set up by the FUS.

An RMS shall populate the elements of the schema with the information to instruct the FUS to set up the entries in the announcement service and configure the corresponding delivery services (multicast or unicast).

Consistent information shall be carried in any CPE management messages appropriate to the firmware update. In this RMS-FUS mode the FUS shall use information provided by the RMS in preference to that provided directly by the CE manufacturer to set up either the announcement messages or the firmware deliveries.

The XML metadata schema is defined in annex B.

The delivery of the metadata is out of scope of the present document.

6.5 Interface 5 - Multicast delivery of firmware update file to network

Two different protocols are defined for the multicast firmware upgrade delivery:

- FLUTE.
- DSM-CC.

The FUS shall support at least one of these protocols for multicast delivery over IP networks. CPEs should implement appropriate protocols for the markets into which they are deployed.

6.5.1 Payload coding and format

Firmware update files which were originally provided by the CE manufacturer and prepared for delivery in the FUS shall be delivered to the population of CPEs through this interface. The format of the files is defined by the CE manufacturer.

6.5.2 Security

The content (firmware update) should be protected by the CE manufacturer to avoid unauthorized use of such software.

The transport stack for the interface is, in the absence of secure message exchange, the FLUTE:UDP:IP or DSM-CC:UDP:IP stack. Since the transport is multicast, the authentication of the message exchange is problematic. The defence for the interface is the authentication of the package (or file) which is the result of the message exchange although the methods for this are out of scope of the present document.

6.5.3 Delivery protocol

The recommended protocols shall be equivalent to existing content delivery protocols and those specified within developing IPTV standards.

A DVB compliant FUS may implement packaging of the image and the use of an additional signature, for example as defined in TR-069 [6]. The metadata contains the elements to indicate whether packaging and signing is in use.

6.5.3.1 FLUTE

This firmware download method is based on the FLUTE protocol [20] and derived from the DVB IP Datacast over DVB-H Content Delivery Specification [37], clause 5. The dynamic file delivery and Raptor FEC scheme defined in [37] are not supported.

FLUTE is built on top of the Asynchronous Layered Coding (ALC) protocol instantiation [21]. ALC potentially combines the Layered Coding Transport (LCT) building block [22], a congestion control building block and the Forward Error Correction (FEC) building block [23] to provide congestion controlled reliable asynchronous delivery of content to an unlimited number of concurrent receivers from a single sender. Congestion control, as defined in [21], is not appropriate in a managed network environment, and thus congestion control is not used for RMS/FUS firmware download. FLUTE is carried over UDP/IP, and is independent of the IP version and the underlying link layers used. FEC shall not be used, because of the relative small size of the delivered files and the availability of other error recovery methods as specified in the present document such as the unicast firmware download interface.

6.5.3.1.1 Profiling of FLUTE file download mechanism in DVB IPI Firmware Update Service

In the present document the term "file" is used for all objects carried by FLUTE (with the exception of the FDT Instances).

RMS/FUS clients and servers shall implement all the mandatory parts of the FLUTE specification, as well as ALC and LCT features that FLUTE inherits. In addition, several optional and extended aspects of FLUTE, as described in the following clauses, shall be supported.

Segmentation of files shall be provided by a blocking algorithm (which calculates source blocks from source files) and a symbol encoding algorithm (which calculates encoding symbols from source blocks).

The use of a single FLUTE channel for a FLUTE session shall be supported.

The use of multiple FLUTE channels for a FLUTE session may be supported by CPEs and FUS servers. For CPEs that do not support multiple channels, it should be possible for them to receive enough data from the first channel named base FLUTE channel in order to declare the channel as complete. The base FLUTE channel is the channel for which the connection information appears first in the SDP session description file. This implies that FDT instances carried over the base FLUTE channel shall not reference files carried over other channels. CPEs that do not support multiple channels shall ignore all but the base FLUTE channel declaration in the SDP session description file.

Each FLUTE channel of a session may send the data packets at a different rate so that it allows faster reception on higher priority channels.

Only "Compact No-Code FEC scheme" [23] (FEC Encoding ID 0, also known as "Null-FEC") should be used, but depending on the implementation the use of FEC is not prohibited.

The "Algorithm for Computing Source Block Structure" described within the FLUTE specification [20] shall be used.

For simplicity of congestion control, all FLUTE channels shall be fully provisioned by the network/service provider so that no transport layer congestion control is necessary. FLUTE channelisation may be provided by a single FLUTE channel.

Files may be content encoded for transport, as described in [20], using the generic GZip algorithm [24]. CPEs shall support GZip content decoding of FLUTE files.

For Gzip-encoded files, the FDT File element attribute "Content-Encoding" shall be given the value "gzip".

In order to avoid IP-fragmentation (fragmentation of one IP datagram into several IP datagrams to changing link MTUs across an end-to-end system) it is recommended that all FLUTE packets (including ALC/UDP/IP headers and the payload of the packet itself) are no greater in size than the smallest anticipated MTU of all links end-to-end. A maximum size of such packet is 1 500 bytes as recommended in [25]. The overhead of protocol headers should also be considered when determining the maximal size of payload data.

Spanning files over several file delivery sessions is not allowed.

Files downloaded as part of a multiple-file delivery are generally related to one another.

Software update packages may be composed of several files. These files usually have to be downloaded as a group because of the existing dependencies. The reception of all files of the software update package is necessary to perform the software update. The metadata includes a list of the files needed to complete a composite update.

6.5.3.1.2 FLUTE session description in DVB IPI Firmware Update Service

The FLUTE specification [20] describes required and optional parameters for FLUTE session and media descriptors. They are provided by SDP over SAP. Annex D defines the SDP parameters used by RMS-FUS including FLUTE specific parameters.

6.5.3.2 DSM-CC

The DSM-CC firmware download uses the DVB System Software Update service profile as defined in [2]. The firmware updates are delivered via DSM-CC carousels in MPEG-2 transport streams as defined in [3] and [4]. For the FUS case the MPEG-2 transport streams are transported via IP multicast channels with direct UDP encapsulation as defined in [1].

The profiling for the URI format for referencing the DSM-CC download from the FUS announcement services is defined in clause C.4. The profiling defined in annex C for the use of the DSM-CC delivery option of the URI allows for correct carousel when:

- The optional "DVB Service ID" may be used to identify the correct carousel when services are delivered to the CPE within a Multiple Program Transport Stream (MPTS) format.
- Multiple carousels may be used within a single service, this uses the optional "carouselId" parameter of the dvb-mcast URI.

6.6 Interface 6 - Unicast delivery of firmware update file to network

An alternative mechanism from multicast is needed to facilitate file downloads especially in cases where multicast firmware upgrades cannot be completed successfully.

The CPE must be provided with the location (URL) of the file to be downloaded using the unicast protocol.

There are several means to provide the location to the CPE which are not mutually exclusive and depend on the CPE implementation, for example:

- The URL is a CPE factory default configuration.
- The URL is included in a QRC response.
- The URL is included in the multicast announcement.
- The URL is configured by the RMS in the managed CPEs.

The way the CPE is configured for unicast firmware upgrade as well as the way it starts the download (e.g. autonomously or based on events) is out of the scope of this clause.

6.6.1 Payload coding and format

The payload coding and format is up to the CPE manufacturer which provides firmware images to be used by the FUS both for multicast and unicast firmware delivery.

Firmware update files which were originally provided by the CE manufacturer and prepared for delivery in the FUS shall be delivered to the population of CPEs through this interface. The format of the files is defined by the CE manufacturer.

6.6.2 Security

The content (firmware update) should be protected by the CE manufacturer to avoid unauthorized use of such software.

Authentication of the server and the client is recommended using the methods described in clause 5.4.

The Customer Premises Equipment Wide Area network Management Protocol document (TR-069 [6]), which the present document extends, recognizes HyperText Transport Protocol (HTTP) as the default package transport protocol, with Secure HyperText Transport Protocol (HTTPS), File Transfer Protocol (FTP) and Trivial File Transfer Protocol as options.

6.6.2.1 HyperText Transport Protocol (HTTP)

To authenticate the server, the CPE and the FUS should support the HTTP:TLS:TCP:IP stack.

The CPE should support authentication of the package (or file) itself.

6.6.2.2 File Transfer Protocol (FTP)

To authenticate the server for the message exchange, the CPE and the FUS should support the FTP:TLS:TCP:IP stack [34] and [35].

The CPE should support authentication of the package (or file) itself.

6.6.2.3 Trivial File Transfer Protocol (TFTP)

There is no specification which covers only the TFTP:TLS:TCP:IP stack. If the CPE is to authenticate the server of the download, the CPE should adopt the same pragmatics as for the FTP:TLS:TCP:IP stack.

The CPE should also authenticate the package or file as required.

6.6.3 Delivery protocol

The present document will follow TR-069 [6] in terms of used protocols: HTTP, HTTPS, FTP, SFTP and TFTP. The CPE shall support both HTTP and HTTPs, which are optional for the FUS.

A CPE must support the use of SSL / TLS and must use SSL / TLS when the file location is specified as an HTTPS URL. The CPE also support both HTTP basic and digest authentication FUS. The specific authentication method is chosen by the file server by virtue of providing a basic or digest authentication challenge.

Authentication techniques are described in clause 5.4.

6.7 Interface 7 - Firmware Update announcement service

The announcement service is identified as interface 7 in figure 5.

The announcement is used to enable the CPE to identify the appropriate firmware update files, if any, which are available for that device, the download methods and transport protocols which should be used.

Each multicast announcement message delivered by the FUS may contain one of the following types of announcement, targeted for different population of CPE:

- Pointer announcement - carrying redirection to another multicast announcement message. The target CPE has to monitor the specified announcement to discover its content.
- Update announcement - carrying metadata based on the schema specified in annex B or SDP attributes specified in annex D. The target CPE is able to start the software update by using information provided by the update announcement type.
- Query announcement - carries redirection to the query/response channel for an unmanaged device. The target CPE has to query the FUS using the query response interface to discover if there is a new software available and the location where to immediately start the download from (there are no restrictions concerning the location provided by the query response interface).

Note that the pointer messages serve a similar function to scan descriptors in DVB SSU, i.e. a flexible high level method of targeting and redirection.

Also note that although the process of looking for an update may be carried out, there may not always be an appropriate update to be downloaded and installed. The time and frequency of the search is not standardised and shall be determined by the CPE or the RMS management policy.

An initial locator for an appropriate entry point into the announcement service must be available to the CPE during the reboot process. This may be provided as part of the bootstrap process, over the CPE management channel at power-on or reboot or be a factory default already embedded in the CPE.

The transport protocol for the multicast announcement shall use at least one of the forms below:

- SAP [27] using the SDP [26] transported via SAP [27] as defined in clause 6.7.2.
- XML based on the schema defined in the annex B transported via DVBSTP as defined in clause 6.7.3 of [1].

6.7.1 Download discovery navigation using the multicast announcement message service

The process for navigating from the entry point supplied by the boot process, the RMS or as a periodic event in the programmed behaviour of the CPE is shown in figures 11 and 12.

Figure 9 shows the stages for the multicast navigation flow options as a result of power on/reboot, the entry points for those options are:

1) Boot sequence for managed devices (inventory available).

2) Boot sequence for unmanaged devices (no inventory available).

3) Autonomous boot sequence based on a factory default address for managed devices by triggering RMS.

4) Autonomous boot sequence based on a factory default address for unmanaged devices.

5) RMS enforcement at boot time.

Note that entry points 1 and 2 both originate from the CPE Bootstrap discussed in clause 4.5 using the FUS Stub file specified in the DVB-IPI Handbook [1], clause 9.

Figure 10 shows the stages for the multicast navigation flow options during the running state, the entry points for those options are:

1) RMS enforcement either during running state.

2) Initialised by autonomously based on a factory default address for managed devices being used to trigger RMS.

3) During running state initiated by an autonomous connection by the CPE to multicast announcement service.

The RMS options shown in figures 7 and 8 (A, B, C and D) are as follows:

- RMS option A: provides URL of service carrying the specific download for CPE to be updated.
- RMS option B: provides URL of service carrying multicast update announcement message which describes specific download for CPE to be updated.
- RMS option C: provides URL of service carrying multicast update pointer message by which the RMS directs the CPE through the same route as that used by unmanaged CPEs searching for an update during steady state.
- RMS option D: unicast RMS address is supplied from an embedded factory default in CPE firmware, RMS then uses one of options A, B or C.

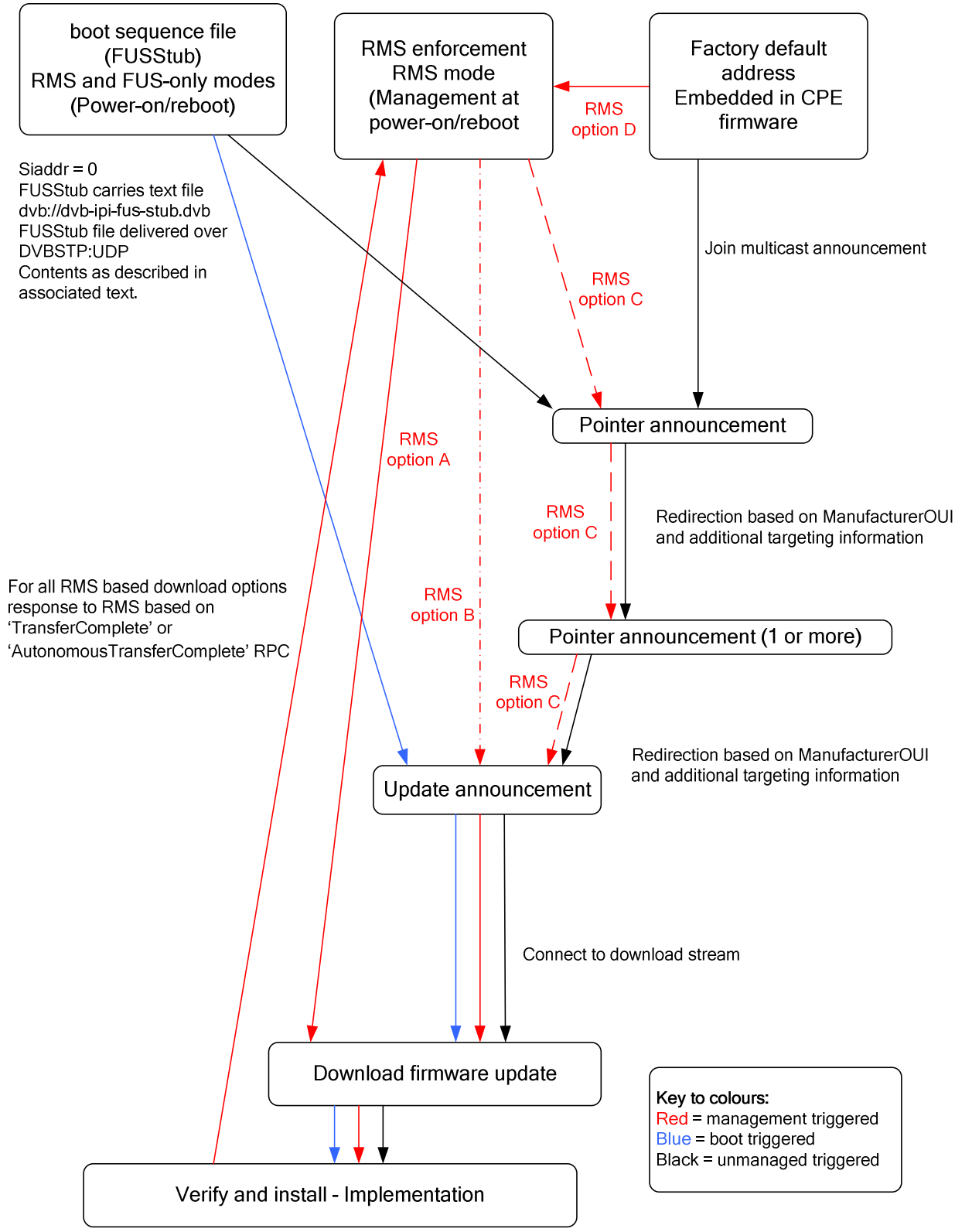

Figure 7: Multicast navigation flow options as a result of power on/reboot

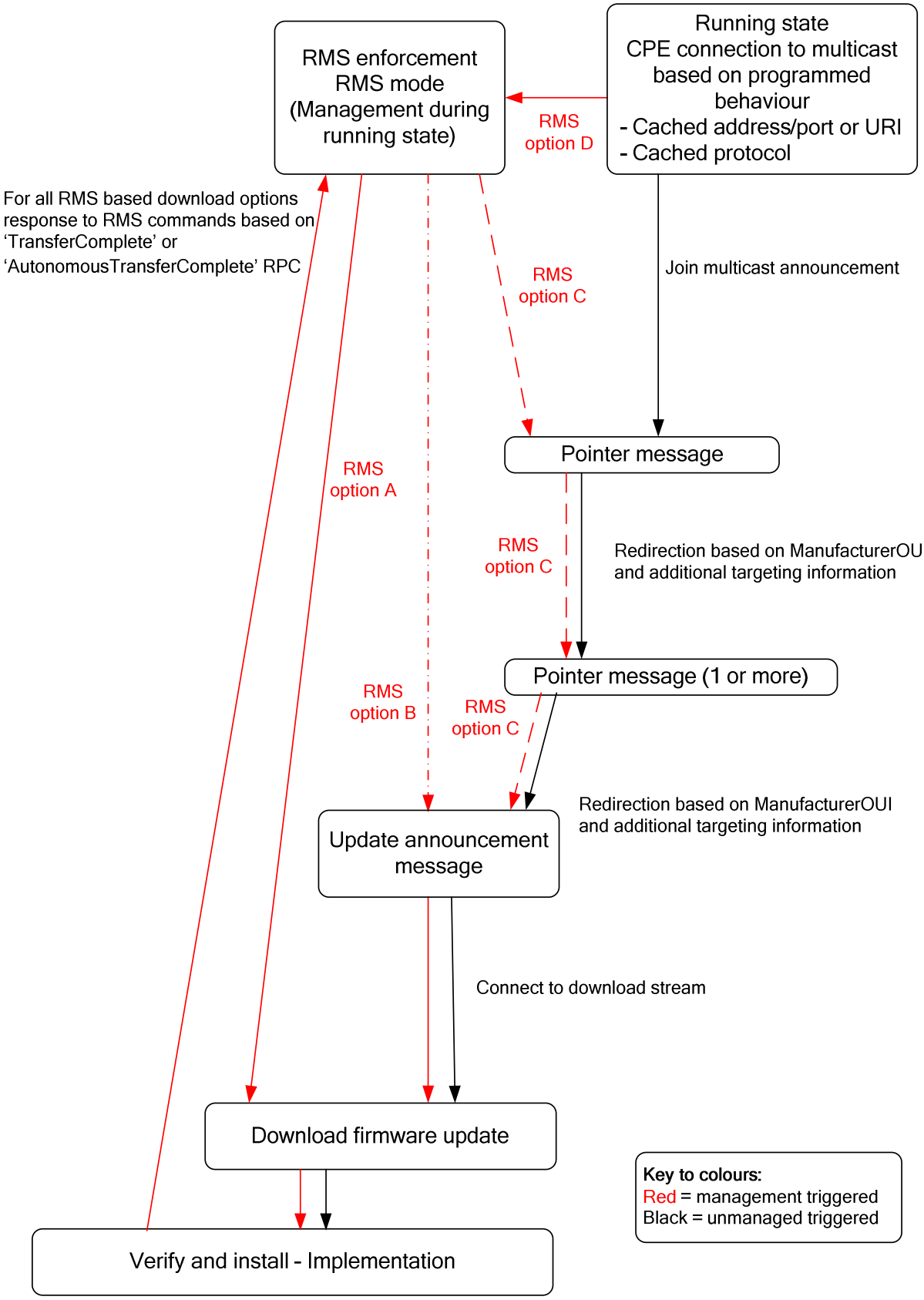

Figure 8: Multicast navigation flow options during running state

6.7.2 Carriage of multicast announcement over SAP transport

The announcement message is based on coding and profiling a Session Description Protocol [26] message which is transported as Session Announcement Protocol [27] payload, therefore this clause does not describe the whole SAP and SDP protocols but only the profile which shall be used for the purpose of multicast announcement.

Each SDP announcement message may contain descriptions of multiple firmware update files, each described as a media sections, each of them is targeted to a population of CPE belonging to a particular CE manufacturer and CPE description.

A specific version of an announcement message document should be contained in a single IP packet.

6.7.2.1 Coding of Session Announcement Protocol

The Session Announcement Protocol (SAP) fields shall be populated as below in table 6. The function of the SAP message is to announce and describe the characteristics of the session within which the update descriptions are carried.

Table 6: DVB RMS specific SAP coding

Session announcement fields		
Field	**Entry**	**DVB-IPI RMS-FUS profiling**
v	"1"	Fixed value, same for SAPv1 and SAPv2 if compatible fields supported
a	"0" = IPv4 "1" = IPv6	Required, set to "0", see note 1
r	"0"	Set to "0", to be ignored by listeners
t	"0" = session announcement "1" = session deletion	Set to "0" for active message Set to "1" only to actively delete session, see note 2
e	"0" = SAP payload unencrypted "1" = SAP payload encrypted	Set to "0" for globally scoped, see note 3
c	"0" = SAP payload uncompressed (zlib) "1" = SAP payload compressed	Compression may be used for messages carrying metadata payloads, see notes 4 and 5
auth len	Length of authentication data	Zero if no authentication data, otherwise = length in 32 bit words
msg id hash	16 bit word	Used to make unique identifier, must not be set to zero, see note 6
originating source	IPv4 or IPv6 IP address	Unicast address of service host, IPv4 only
authentication data	Certificate, <=1k in length	Recommended
payload type	"application/sdp""/0"	Required
NOTE 1: DVB-IPTV Handbook currently only supports IPv4. NOTE 2: The options describing the circumstances under which the CPE should close a session are described in [27]. NOTE 3: Encryption of payload is not recommended on globally scoped address ranges, however, it may be appropriate for some applications using administratively scoped sessions. NOTE 4: If compressed payload, compression must be done before encryption. NOTE 5: If used compression will be applied to the full message after the payload type in the SAP header. NOTE 6: A change in the message hash indicates that the payload has been updated.		

6.7.2.1.1 Limitations

Due to the structure of the SAP encoding [27] there is no provision for carrying a message type identifier field in the SAP header, which would be available without parsing the SDP payload.

6.7.2.1.2 Authentication method to be used in SAP payload

The authentication data in SAP is defined in RFC 2974 [27] as shown below in figure 9, its use is optional for µ RMS-FUS.

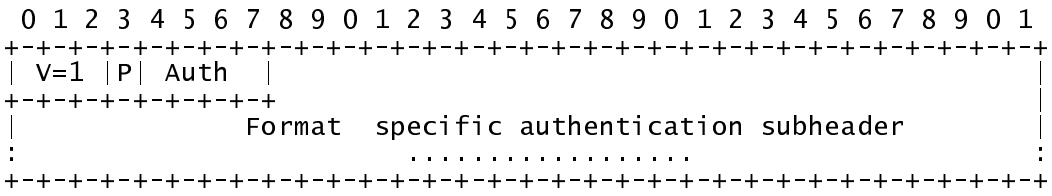

Figure 9: Format of the authentication data in the SAP header

If used the FUS profile for signing SAP announcements for firmware update is as follows:

- The Version Number MUST be set to 1 as indicated in RFC 2974 [27].
- The Authentication Type MUST be set to 1 indicating CMS format as defined in [32] is to be used.

The format specific authentication subheader will have exactly one CMS SignedData object with the fields provided as follows:

ContentInfo {

contentType	id-signedData, -- (1.2.840.113549.1.7.2)
content	SignedData

}

SignedData {

version	shall be set to 3 indicating use of CMS
digestAlgorithms	DigestAlgorithmIdentifiers, there shall be only one
encapContentInfo	EncapsulatedContentInfo, this shall not be present (see note)
certificates	CertificateSet, carrying the signer certification path
crls	CertificateRevocationLists, this shall not be present
signerInfos	Set of SignerInfo, only one is allowed

}

SignerInfo {

version	set to 3, indicating use of CMS
sid	SignerIdentifier
digestAlgorithm	DigestAlgorithmIdentifier, this shall be set to" SHA-1"
signedAttrs	SignedAttributes, this shall be present
signatureAlgorithm	SignatureAlgorithmIdentifier,
signature	SignatureValue,
unsignedAttrs	UnsignedAttributes, these attribute fields are optional

}

The encapContentInfo element MUST be empty, since this is an external signature and the message data resides outside the signature itself. The signed content consists of the SDP payload of the announcement message.

6.7.2.2 Payload coding of SDP header

The SDP structure describes the identifiers and characteristics of the actual update description session.

A single SDP announcement session type is defined, which can be identified by a session name defined by DVB ("dvb-update"), shall be used to describe all the announcement message types. The session name can be used as an initial filter to identify messages carrying update announcements intended for DVB IPTV CPEs within the internet scoped multicast service.

Table 7 describes the profiling of the session-description header fields as required in the DVB RMS application. The fields not explicitly described shall not be used in DVB RMS applications to characterise these messages.

Table 7: DVB RMS-FUS specific SDP coding

Session description fields		
Type	**Value**	**DVB profile**
v=	0	Fixed value.
o=	<username> <sess-id> <sess-version> IN IP4 <unicast-address>	Mandatory, this string represents a globally unique combination identifying the session. For the DVB RMS application this field shall be made up as specified in [26] clause 5.2, with the following restrictions: The combination of <username><sessionid> uniquely identifies a session to a given target population. This combination is referred to as "RMS-FUS_session_ID" when used in profiling the dvb-mcast URI for RMS-FUS The FUSInfo.Name as used in the metadata is used to populate the <username> field. If provided, the <unicast-address> field will contain the unicast source address the FUS uses for the announcement.
s=	dvb-update	Fixed value used only for DVB RMS-FUS.
i=	<session-information>	Optional, if present may be coded as the OUI or name of the agency responsible for update.
b=	CT:<session-bitrate>	Optional, if present should include all updates described within this session (bandwidth per connection).
t=	<start-time> <stop-time>	Optional. Values StartTime and StopTime from <FirmwareUpgradeInfo. ValidityTimeRange> in the metadata are use to populate <start-time> and <stop-time>. This time range shall apply to all multicast sessions in the media description part of the announcement. StartTime = should indicate when multicast session starts. StopTime = time when session will end. The session will be stable within this period.
a=	<attributes…>	This session-level DVB specific attribute is required, and has the form: x-dvb-rms-ce-manufacturer:<ManufacturerOUI> <ManufacturerOUI> It is a sequence of one or more ManufacturerOUIs associated with those for which information is carried in the media-description sections following the session-description. The CPEs may use this session-level attribute to check if this announcement message is from its CE manufacturer before to start parsing all the following media-descriptions.

The session description part of the message may be used by the CPE to identify the announcement source and the CE manufacturers to which the following media-descriptions refer to.

The "x-dvb-rms-ce-manufacturer" attribute is specific to the DVB RMS application and is defined in the present document in annex D.

EXAMPLE:

```
v=0
o=dvb-fus 2890844526 2890842807 IN IP4 10.47.16.5
s=dvb-update
t=2873397496 2873404696
a=x-dvb-rms-ce-manufacturer:00D09D 00D09E 00D09F
...
```

This example shows an SDP message sent by the FUS named "dvb-fus" having IP address 10.47.16.5. The message carries announcements for the manufacturers which OUI are 00D09D, 00D09E and 00D09F. The rest of the message is omitted in this example for sake of simplicity.

One or more media sections will follow the SDP message header (session-description). Each media section may have specific attributes ("a=<…>") associated with it, those specific to DVB are defined in annex D.

6.7.2.3 Coding of media sections

Session Description Protocol (SDP) coding will be used for SAP multicast announcements as defined in the present document.

The mechanisms for describing the payload are specified in [26] and profiled in clause 6.7.2.2.

SDP uses media descriptions and attributes to describe the media sections, DVB will use those which are already standardised where appropriate and will define some DVB specific attributes for the purposes of the present document, the DVB specific attribute are defined in annex D.

The same SDP message may contain more than one media description, each carrying information for either a specific population of CPEs, or for multiple FLUTE channels for a single population of CPEs.

The attributes will be used either to carry the parameters of the announcements which make up the media sections. A media section will be headed by a media description and may contain multiple attributes as the section payload but will have a single target population of CPEs. If multiple attributes are present in a media section the targeting they describe must be considered in a "logical AND" mode.

Annex D also includes some examples of the media sections described.

6.7.2.4 IP address range options

Since this service may have to operate in an open internet environment no specific restrictions are placed on the IP address ranges in the present document (additional to those in [27]) but it may be possible to consider optimisations to this problem in the longer term.

It is therefore possible to use either the global or the administratively managed IP address options described in [27].

6.7.2.5 Security

Encryption of the SAP messages is not supported.

Authentication of the message exchange is recommended as defined in clause 6.7.2.1.2.

Client authentication is not specified.

6.7.3 Carriage over DVBSTP

DVBSTP can be used as one of the alternative methods of delivering the announcement messages, this clause describes the coding of the DVBSTP transport session header to carry out this task.

The description of the DVBSTP coding should be assumed to be informative, with TS 102 034 [1], except where specific profiling of the fields is used for this application.

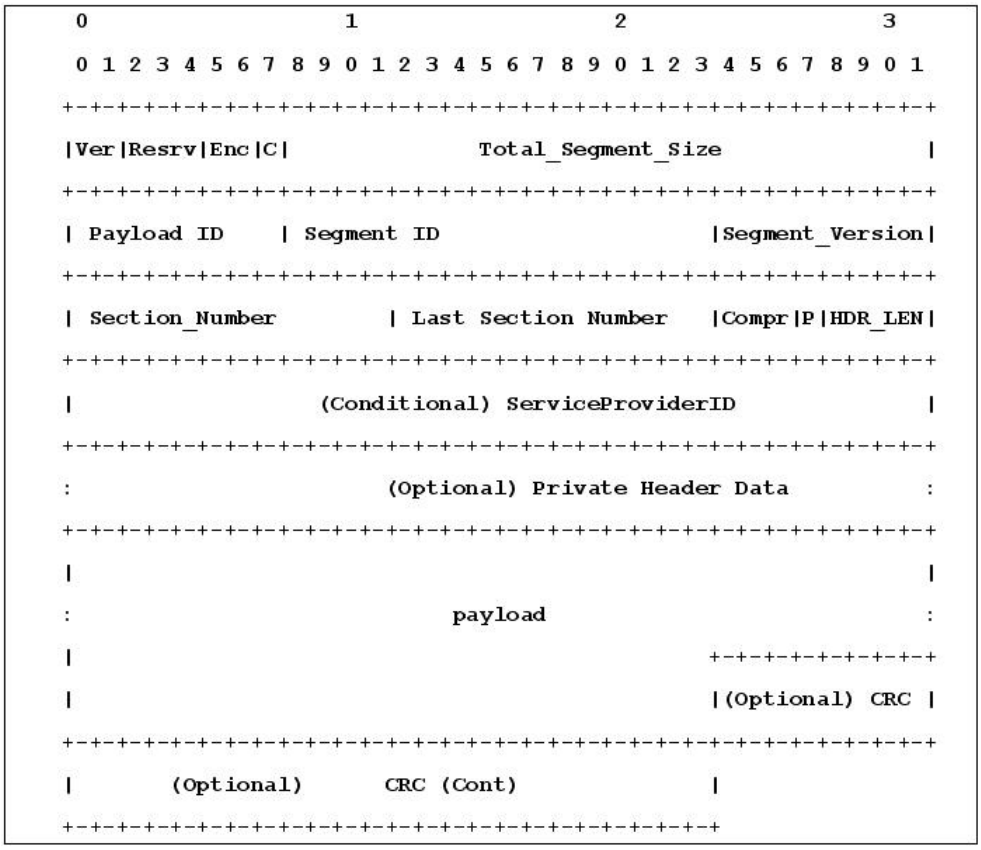

Figure 10: DVBSTP packet header structure

6.7.3.1 Coding of DVBSTP

When the FUS announcement discovery information is carried using multicast delivery over a DVBSTP:UDP:IP transport the packet protocol defined in this clause shall be used. All values defined below shall be transmitted in normal IP network byte order (most significant byte first).

Syntax

Figure 10 shows the DVBSTP packet header structure, this is shown as informative with the definitive version being in [1].

When used for carrying announcement messages the DVBSTP fields shall be coded as in table 8.

Table 8: Profiling of DVBSTP for RMS-FUS

Field	Value
Ver	00
Resrv	000
Enc	00
C	0 if no CRC carried, 1 if CRC carried, This flag may only be set on the final packet in a segment, i.e. when section_number is the same as last_section_number.
Total_segment_size	The cumulative size of all the payloads of all the sections comprising the segment (i.e. ignoring headers and CRC, if present).
Payload_ID	Set to 0xB2, (see table 1 of [1]).
Segment ID	Value used to identify a segment of data for the declared payload type 0xB2.
Segment version	Used to define the current version of the segment being carried.
Section number	Field identifying the number of this section. The first section in a segment shall be 0.
Last Section number	Which specifies the last section number in a segment.
Compression (Compr)	See table 10 of [1], defining compression values. (Note that all segments of a given payload ID shall share the same compression value.) Set to "0b000" for no compression, 0b010 for GZip.
ProviderID flag	Flag signalling if the ServiceProviderID field is present. The value "1" defines the presence of the ServiceProviderID field in the header.
Private Header length	Length of private data.
ServiceProvider ID	32-bit number that is used to identify the service provider. This number shall be an IPv4 address, as detailed in clause 5.4.1.3 of [1].
Private Header Data	This is private data. The meaning, syntax, semantics and use of this data is outside the scope of the present document. This field shall be a multiple of 4 bytes.
Payload	Message data as defined in clause 6.7.3.2.
CRC	optional 32-bit CRC. See clause 5.4.1.2 of [1] for details.

6.7.3.2 Payload coding for DVBSTP

The present document extends [1] in that the service over DVBSTP shall carry the RMS-FUS announcement messages, which consists of the RMS-FUS XML metadata schema which identifies the firmware updates. The RMS-FUS XML metadata carried as payload is described in annex B.

Different announcements may be carried on separate segments over the same IP address.

6.7.3.3 IP address options for DVBSTP

TS 102 034 [1] places no restrictions on the addressing, the present document will follow that specification in that sense.

6.7.3.4 Security

Message encryption and authentication is not supported by DVBSTP.

Neither client nor server authentication is specified.

6.8 Interface 8 - Query/response interface from FUS to home environment (QRC)

The query response channel (QRC) is a SOAP 1.1 [12] (a standard XML syntax based protocol used to encode remote procedure calls) based interface where the server is the FUS and the client is the CPE.

The interface is mandatory in the FUS for the FUS-only implementation but not required for the RMS-FUS mode implementation. It is optional for the CPE implementations.

All elements and attributes defined as part of this version of the QRC interface are associated with the namespace identifier "urn:dvb-org:qrc-1-0".

6.8.1 Download discovery navigation using the unicast query/response channel

The process for navigating from the entry point supplied by the boot process or as a periodic event in the programmed behaviour of the CPE is shown in figure 11 using the query/response channel as the method of obtaining the information about the appropriate update.

Figure 11 shows the stages for the unicast navigation flow options as a result of power on/reboot, the entry points for those options are:

1) Boot sequence file (FUSstub) supplies address of query/response channel in FUS.

2) Query/response channel address provided as part of factory default configuration in CPE and initiated as an autonomous action at boot time (independent of the boot sequence).

3) During running state initiated by an autonomous connection by the CPE to the unicast query/response channel.

6.8.2 Payload coding and format

The remote procedure call (RPC) mechanism offered by SOAP 1.1 [12] is used for the bi-directional communication between the CPE and the FUS and the communication is always initiated by the CPE by using one of the methods specified in this clause.

Communication between CPE and FUS is stateless. Therefore each response only depends on the invoking method, because it is not required to the FUS to maintain CPE information between multiple method invocations.

The present document is intended to be independent of the syntax used to encode the defined RPC methods. The particular encoding syntax is compliant with the SOAP implementation requirements.

The procedure name, followed by their response name is:

- FirmwareUpdateCheck, FirmwareUpdateCheckResponse.

The calling arguments and the semantics for the remote procedure call are defined in tables 9, 10 and 11.

6.8.2.1 FirmwareUpdateCheck/FirmwareUpdateCheckResponse

The CPE invokes FirmwareUpdateCheck to ask the FUS whether new firmware is available for download, if no URL is returned it means that the CPE already has the latest available firmware version installed. By using the information in the RPC the FUS is able to narrow the choices of possible responses to the CPE.

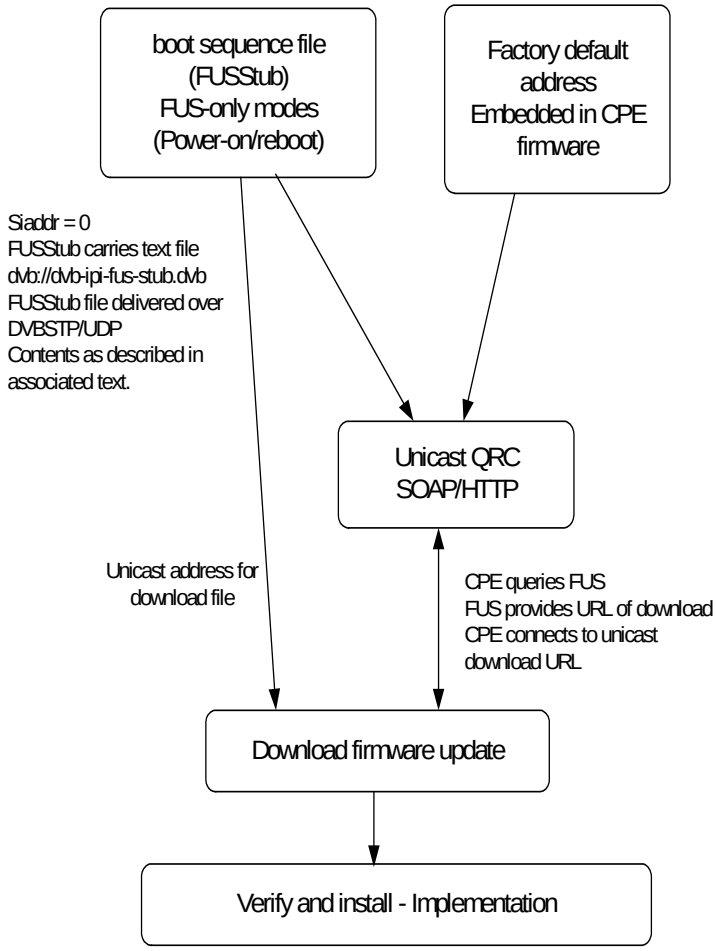

Figure 11: Navigation flow options at boot using unicast messaging

Table 9: FirmwareUpdateCheck arguments

Argument	Type	Description
ManufacturerOUI	String	Organizationally unique identifier of the device manufacturer. Represented as a six hexadecimal-digit value using all upper-case letters and including any leading zeros. The value MUST be a valid OUI as defined in [10].
ProductClass	String	Identifier of the class of product for which the serial number applies. That is, for a given manufacturer, this parameter is used to identify the product or class of product over which the SerialNumber parameter is unique.
SerialNumber	String	Optional. Identifier of the particular device that is unique for the indicated class of product and manufacturer.
MACAddress	String	Optional. MAC address of the device.
HardwareVersion	String	Hardware version of the device.
SoftwareVersion	String	Software version of the device.
VendorSpecificInfo	String	Optional. This string contains some vendor specific information. This optional string could contain, for example an array of URL (multicast or unicast) the CPE has previously used to download the firmware but without successfully completion of the update.

Table 10: FirmwareUpdateCheckResponse arguments

Argument	Type	Description
SourceURL	ResourceAccessInfoType[]	Array of ResourceAccessInfoType objects (table 14), as described in the metadata. Each item in the array specify a choice of source file location. It is an ordered list where the first item has the highest preference and the last item has the lowest preference from the FUS. All mandatory protocols for interfaces 5 (multicast), 6 (unicast) of the present document must be supported. The array can be empty in case no meaningful answer is available from the server.

Table 11: ResourceAccessInfoType arguments

Argument	Type	Description
URL	String	URL (as defined in [31]) specifying the source file location.
Username	String	Username to be used by the CPE to authenticate with the file server. This string is set to the empty string if no authentication is required.
Password	String	Password to be used by the CPE to authenticate with the file server. This string is set to the empty string if no authentication is required.

6.8.3 Security

The CE manufacturer and FUS shall use a standardised mechanism to authenticate each other before an operation is performed for either metadata (interface 2) or firmware update through this interface.

Since the QRC interface is defined over the SOAP 1.1 [12] standard protocol, which is carried over HTTP packets, a secure channel may be created between CPE and FUS by using HTTPS.

The set of requirement specified in WS-I Basic Security Profile 1.0 [40] must be satisfied by QRC implementations in order to guarantee the right security level and the maximum interoperability between clients and servers.

The Basic Security Profile 1.0 [40] (clause 3.1.1) prohibits use of SSL 2.0, therefore SSL 3.0 or TLS 1.0 ([16], and [34]) may be supported by the CPE and must be used when the FUS location for the QRC interface is specified as an HTTPS URL. The CPE must also support both HTTP basic and digest authentication, and the specific authentication method is chosen by the file server by virtue of providing a basic or digest authentication challenge as required by HTTP.

The way the CPE is configured and shares the credentials with the FUS is out of the scope of this clause.

6.8.4 Delivery protocol

SOAP is the protocol which must be used to deliver messages between the CPE (SOAP client) and the FUS (SOAP server) for the QRC interface. The SOAP implementations must adhere to the interoperability requirements defined in the WS-I Basic Profile 1.0 [41] and WS-I Basic Security Profile 1.0 [40] in order to guarantee the maximum interoperability, therefore SOAP 1.1 [12], XML 1.0 [11] and HTTP/1.1 [14] have to be considered as reference specification.

The set of requirements specified in WS-I Basic Profile 1.0 [41] must be satisfied by QRC implementations in order to guarantee the maximum interoperability, therefore SOAP 1.1 [12], XML 1.0 [11] and HTTP/1.1 [14] shall be considered as the reference specification.

Note the CPE (SOAP client) needs only to implement the minimum request/response message exchange behaviour in order to keep the implementation as simple as possible: it is not mandatory for the CPE to implement the entire SOAP protocol stack.

The interface is mandatory in the FUS for both FUS-only implementation but not required for the RMS-FUS mode implementation. It is optional for the CPE implementations.

The service provider should consider the scalability issues associated with implementing QRC within its network. The operation of QRC within any network is optional.

6.8.5 Message back-off and retry strategy

In the event of contention when using unicast messages over the query/response channel (interface 8) some specified back-off and retry behaviour is necessary, as an example this may be a problem if a large population of CPEs all boot up at the same time and send "Inform" messages to the RMS.

The strategy which unmanaged CPEs shall implement will be the similar to that described in clause 9.1.2.1 of the DVB IPTV Handbook [1]. The exception to that specification will be that a granularity of 1 second time units should be used.

An initial back-off of 1 second shall be set for the back-off timer each time the CPE attempts to send a unicast message to the server. Immediately before each attempt to establish a connection, a random delay of between back-off and 2*back-off seconds shall be imposed. Upon failure to establish this connection, the back-off timer shall be doubled and the connection will be retried. When doubling of the back-off timer results in an arithmetic overflow, retry attempts should be abandoned.

If the use of the QRC is enabled for managed CPEs the back-off policy defined by the configuration required by the RMS shall be used.

6.9 Interface 9 - CPE management interface

This interface shall support the functionalities described in clause 5.2.3, for the bi-directional communications between the RMS and the CPE.

In annex A, the protocol TR-069 for remote management is profiled for purposes of interface 9.

Other remote management methods may also be used with the present document, although the issues associated with their implementation are not considered in the present document.

6.9.1 Management interface requirements

For TR-069 the message set (RPCs) for the Customer Premises Equipment (CPE) and the Remote Management Service (RMS) are described in clause 3.6 of TR-069 [6].

6.9.2 Security

The present document recommends SOAP encoded over HTTP:TLS:TCP:IP for secure message exchange. The ciphersuite recommendations are described in the security section of the present document. This is in alignment with TR-069 [6] as described in annex A.

6.9.3 Delivery protocol

The present document recommends the SOAP over HTTP:TLS:TCP:IP stack for secure message exchange, compliant with [12]. Again this is in alignment with TR-069 [6] as described in annex A.

6.10 Interface 10 - RMS management and control

This is a business to business interface which is out of scope for the present document.

6.11 Interface 11 - FUS control interface and file management interface

This is a connection which is internal to the FUS and is out of scope for the present document.

6.12 Interface 12 - Update file streaming to delivery formatter

This is a connection which is internal to the FUS and is out of scope for the present document.

6.13 Interface 13 - RMS device registration

This is a connection which is internal to the RMS and is out of scope for the present document.

7 Data Model

The data model objects needed to define the requirements for and the management of DVB Content Download Service (CDS) have been developed and are described in annex E. These data model objects are generically applicable to any remote management data model.

Although in principle a number of CPE management protocols might be used across Interface 9, TR-069 [6] as defined by Broadband Forum (BBF) is the most popular protocol for IP consumer device management. Consequently the usage of the BBF methods has been investigated in more detail for the purpose of managing CPEs especially including support of DVB specific services.

The BBF technical reports relevant for IPTV CPE management comprises:

- TR-069 [6] defining the generic protocol for CPE to RMS bidirectional communication.
- TR-106 [7] defining the generic device data model.
- TR-135 [8] defining the IPTV CPE specific data model.

The extensions to the TR-069 / CWMP framework required to support the DVB RMS-FUS capabilities as described in the present document are specified in annex A.

The extensions to TR-069 / CWMP framework required for supporting DVB CDS requirements are specified in annex F. The DVB CDS provides services for the download of content items to a local storage of a CPE via a broadband IPTV connection and is specified in [1].

Extensions to TR-069 and TR-106 are not necessary. The extensions for the CPE data model are based on TR 135 Issue 1 [8]. The following requirements have been taken into account when specifying the necessary extensions for the data model:

- The CDS shall enable management and use of storage.
- The CDS shall support business models based on user managed and/or operator managed local storage.
- The CDS may allow operators to query the content items they have delivered to a particular user's local storage.

Annex A (normative): Use of TR-069 / CWMP for DVB RMS-FUS

NOTE: This annex contains normative information if the firmware update service is applied in association with the Broadband Forum methods as described in TR-069 [7].

This annex provides an overview of the Broadband Forum TR 069 framework for remote management at interface 9 (RMS-Device, see figure 5) of the RMS-FUS system. Examples of how the TR-069 / CWMP protocol could be used for implementing the CPE Management interface functionalities are also provided. Any conflict between this annex and the normative version of TR-069 [6], as reported on the Broadband Forum site (http://www.broadband-forum.org), shall be resolved in favour of the normative BBF version.

This version provides a view of the changes adopted by BBF in its TR-069 [6] framework to support the DVB RMS-FUS functionalities. These changes comprise those made to the TR-069 protocols, now contained in TR-069 [6], and the new objects required by DVB RMS-FUS, contained in TR-157 [9].

This annex describes how the functionality described in the body of the present document can use the Broadband Forum TR-069 [6] methods. A DVB RMS which uses some or all of the methods described in TR-069 [6] is therefore a full or partial instantiation of the ACS as specified in TR-069 [6]. In this annex the term "RMS" will be used since this is a document scoped and specified by DVB.

A.1 Reference specifications within the TR-069 framework

The TR-069 framework includes the specification of the CWMP remote management protocol (TR-069 [6]) and the associated data models. These cover IPTV as well as other functionalities of the Digital Home. For IPTV the TR-069 framework documents of interest are:

- TR-069 [6], containing the CMWP protocol extensions for managing multicast downloads.
- TR-106 [7], containing a collection of requirements common to all TR-069 data models. It also contains the specification of the Device data model object and a number of objects that should be common in CPE implementations. All these data models are expressed in xml format. This is based on the XML data model schema included in TR-106.
- TR-157 [9], containing a collection of data model components (objects / object sets that implement a specific functionality and thus should be used atomically) for use by any TR-069 CPE. The component that configures CPEs to support the autonomous download specified by RMS-FUS is included here.
- TR-135 [8], containing the data model for a TR-069 STB for IPTV and other forms of Digital Television.
- TR-140 [5], containing a data model for a TR-069 Storage Service.

A.2 CWMP Remote Procedure Calls

CWMP as described in TR-069 [6] is the protocol for remote management of Home Network CPEs specified by the Broadband Forum. This is mainly a SOAP based collection of RPCs (Remote Procedure Calls), issued partly by the RMS and partly by the CPE.

Concerning RMS-FUS, the RPC Method Specification (see annex A of TR 069) defines a mechanism to facilitate file downloads or (optionally) uploads for a variety of purposes, such as firmware updates or vendor-specific configuration files. File transfers can be performed by means of Unicast (for downloads and uploads) or Multicast (for downloads only) transport protocols. Unicast protocols include HTTP/HTTPS, FTP, SFTP and TFTP. Multicast protocols include FLUTE and DSM-CC. Support for HTTP/HTTPS is mandatory, and protocols other than those listed here can be supported.

File transfers can be initiated by either side. These are implemented by means of the CWMP RPCs described in the following paragraphs. RPCs are exchanged between RMS and CPE in CWMP sessions.

TR-069 specifies for each RPC the relevant support requirements (required/optional). According to which of the TR-069 optional RPCs are specified different scenarios can be deployed.

The rest of this clause contains a list of the mandatory and optional RPCs that are relevant in the examples of FUS Management functionalities described in clause A.3 implemented via TR-069. For the complete list of RPCs specified by TR-069 please refer to the TR-069 standard [6].

A.2.1 RPCs required at the RMS and CPE side by TR-069 Framework

The Inform RPC (required both on the RMS and CPE side) is used by the CPE to start a CWMP session with the RMS. The Inform RPC contains Event Codes to describe the reason why the CPE set up the CWMP Session.

The TransferComplete RPC (required on the RMS and CPE sides) is used by the CPE to signal back to the CPE completion of the (multicast or unicast) download requested by the RMS by means of a TR-069 Download RPC.

The AutonomousTransferComplete RPC (required on the RMS side, OPTIONAL on the CPE side) is used by the CPE to signal back to the RMS completion of a (multicast or unicast) download that was NOT requested by the RMS by means of a TR-069 Download RPC. This is the case, for instance, of a CPE performing autonomously a download by getting the advertisement via multicast channel or querying the QRC server.

The Download RPC (required on the RMS and CPE) can be used by the RMS to request the CPE to download and apply a file, including but not limited to a firmware image. The file is specified via a URL: support for HTTP and HTTPS is mandatory, and support for other protocols is optional, including multicast protocols. The RMS can specify a username, a password, and a delay before the download should start.

The GetParameterNames, GetParameterValues and SetParameterValues (mandatory on both RMS and CPE) as well as GetParameterAttributes and SetParameterAttributes RPCs (mandatory on the CPE but optional on the RMS) may be used by the RMS to configure and manage the CPEs.

A.2.2 RPCs optional at the RMS and CPE side by TR-069 Framework

The RequestDownload RPC can be used by a CPE to request a download to the RMS. Upon receiving a RequestDownload the RMS, may as a result call the Download RPC on the requesting CPE.

The SchedfuleInform RPC may be used by the RMS to request the CPE to schedule a one-time Inform method call.

The Upload RPC can be used by the RMS to request the CPE to upload a file. Apart from the direction of the transfer, its operation is similar to the Download RPC.

The GetAllQueuedTransfers RPCs can be used by the RMS to return a list of a CPE's queued downloads and uploads.

A.3 Remote Management functionality using CWMP

A.3.1 How to exploit CWMP for DVB remote management functions

In this clause some examples will be provided of how TR 069 / CWMP can be used to perform the basic remote management tasks listed in clause 5.2.3, regarding both provisioning and assurance (1 to 8) and firmware update (9 to 14) in a managed environment (RMS FUS).

1) RMS and CPE may both initiate or request to initiate the communication for management purposes: all management sessions are initiated on the CPE side by the CPE invoking the Inform RPC on the TR 069 RMS. It is assumed that the CPE discovered the TR 069 URL of its RMS in some way. The RMS can use the ConnectionRequest mechanism in order to ask the CPE to start a new management session, i.e. invoking the Inform RPC.

2) RMS may read the configuration of each managed CPE (getting parameter values): this can be done by the RMS using the GetParameterValues RPC.

3) RMS may change the configuration of each managed CPE (setting parameter values): this can be done by the RMS using the SetParameterValues RPC. An example is that the RMS may configure the CPE to support the ability for downloads to be carried out based on a multicast announcement or QRC query, the CPE must be configured by the RMS by means of objects ".DownloadAvailability" and ".ManagementServer.AutonomousTransferCompletePolicy" contained in TR-157 [9].

4) CPE may send status report to the RMS by setting up a CWMP session every time it is required by the CPE policy. This is done by the CPE sending an Inform to the .RMS. The Inform argument list contains the CWMP device identifier, the firmware version (SoftwareVersion in TR-106), the hardware version, and a list of parameters that changed value after the last Inform., Specific parameters could be added for DVB purposes.

5) RMS may request the execution of diagnostic tests on the CPE and collect the results. By setting some specific parameters with the SetParameterValues command, the RMS is able to initiate various diagnostic tests in the CPE. Test results are collected by the RMS in the Inform that indicates completion of the test, or by explicitly reading specific parameters with GetParameterValues.

6) RMS may invoke operational commands on the CPE, e.g. reboot, factory reset: the CPE must implement the mandatory Reboot RPC and optionally it may implement the FactoryReset RPC.

7) RMS may configure on the CPE the requested behaviour in terms of active/passive/ notification of events/alarms, e.g. for fault management, performance management and SLA management and statistics collection: this can be done by configuring the required passive or active notification behaviour, by means of the SetParameterAttributes and GetParameterAttributes RPCs. Parameters with passive notification behaviour are included in each Inform when their value changes. Parameters with active notification behaviour trigger a new Inform when their value changes. The optional ScheduleInform RPC may also be used by the RMS to request the CPE to send an Inform at some time in the future. The RMS may also define intervals at which the CPE will send an Inform message, e.g. once per day at a regular time.

8) CPE may autonomously send events/alarms to the RMS, in compliance with the configured behaviour: This can be seen as part of the status report already described in item 4.

9) RMS may command the CPE to download and apply a file (e.g. a configuration file or CPE Firmware): by using the Download RPC the RMS can command the CPE to start the download and apply after a specified delay that could be zero.

10) RMS may command the CPE to upload a file. Almost the same as in the previous point but the RMS uses the Upload RPC.

11) CPE may inform the RMS of the success/failure of any commanded file transfer (download/upload): TR-069 Amendment 2 specifies the conditions under which the CPE should notify completion of a file download or upload. This can be done by using the DownloadResponse / UploadResponse, or the TransferComplete RPCs, as specified in [6].

12) CPE may autonomously start a firmware/software update process by explicit query (by issuing a RequestDownload RPC to the RMS).

13) CPE may inform the RMS of the successful/unsuccessful completion of the autonomous firmware/software update process. Where the firmware update has been carried out as a result of use of the multicast announcement or a query to the QRC, this is usually done by means of the Autonomous-Transfer-Complete RPC. Where the firmware update has been carried out as a result of issuing a RequestDownload RPC to the RMS, this is usually done by means of the Transfer-Complete RPC.

14) RMS may implement recovery mechanism to respond to download failures or configuration problems. This has mainly to do with RMS side procedures making use of the TR-069 RPCs described in the previous points.

A.3.2 Examples of firmware update tasks in a managed environment

The following informative examples show how CWMP may be used for DVB purposes in order to perform firmware update tasks in a managed CPE environment (RMS FUS).

EXAMPLE A: Bootstrap/boot of a new managed CPE including its firmware upgrade

- Due to the first time installation, the CPE sends an Inform to its RMS containing the "0 BOOTSTRAP" event code and, as usual, the firmware version among other parameters. In any case the CPE sends an Inform to the RMS every time it re-boots, containing the "1 BOOT" event code.
- The RMS verifies whether the firmware version needs s to be updated and, if an update is necessary, uses the Download RPC to request the CPE to perform the firmware update.
- The CPE performs the firmware update as requested by the RMS.
- After the firmware update has completed, the CPE initiates a new management session by invoking the Inform RPC followed by the TransferComplete RPC to notify the RMS of the success or failure of the update procedure.
- The RMS can perform, especially in case of first time installation, some additional configuration operations on the CPE by using, for example, the SetParameterValues and GetParameterValues RPCs.

EXAMPLE B: Multicast upgrade triggered by multicast announcement

- The RMS configures the FUS to start a new multicast update for a population of CPEs.
- The FUS sends the multicast announcement to the network, with some filtering parameters, as required by the RMS, in order to let only the target CPEs consider the announcement.
- All the CPEs listening for an announcement from the FUS receive the announcement and, depending on the filtering information included in the announcement message, start to listen for the multicast firmware delivery.
- All of the target CPEs perform the multicast update as specified by the received announcement.
- After the multicast update has completed, either successfully or not, each target CPE sends an Inform notification to the RMS followed by the AutonomousTransferComplete message.

EXAMPLE C: Unicast recovery from a multicast delivery failure

- The CPE executes a multicast firmware update, but, after all the possible autonomous fallback/recovery procedures, for some reason the operation definitely fails.
- The CPE sends an Inform notification to the RMS followed by the TransferComplete or AutonomousTransferComplete message containing the appropriate FaultCode argument.
- The RMS then decides to use a more reliable means to update the CPE and sends a Download RPC in order to let it start a new download procedure by using one of the available unicast protocols, as specified by CWMP.
- After this unicast firmware update has completed, either successfully or not, the target CPE again sends an Inform notification to the RMS followed by the TransferComplete message.

Annex B (informative): Metadata schema

NOTE: This annex contains the textual representation of the XML schema available as "DVB-TM-IPI-RMS-metadata phase 2 v10.xsd" associated with the present document.

This annex contains the textual representation of the XML schema available as "DVB-TM-IPI-RMS-metadata phase 2 v10.xsd" associated with the present document.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with Oxygen v10.3 by Dave Walton (Echostar) Aug 2009-->
<xsd:schema targetNamespace="urn:dvb:ipi-rms:phase2:2009-01"
    xmlns:rms="urn:dvb:ipi-rms:phase2:2009-01"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified" attributeFormDefault="unqualified"  version="1.0">
    <xsd:simpleType name="ManufacturerOUIType">
        <xsd:annotation>
            <xsd:documentation>
                Organizationally unique identifier of the device manufacturer.  Represented as a six hexadecimal-digit value using all upper-case letters and including any leading zeros.  The value MUST be a valid OUI as defined in IETF.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:restriction base="xsd:string">
            <xsd:pattern value="(0-9)|(A-F)"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="ProductClassType">
        <xsd:annotation>
            <xsd:documentation>
                Identifier of the class of product for which the serial number applies.  That is, for a given manufacturer, this parameter is used to identify the product or class of product over which the SerialNumber parameter is unique.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
    <xsd:simpleType name="HardwareVersionType">
        <xsd:annotation>
            <xsd:documentation>
                A string identifying the particular CPE hardware model and version.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
    <xsd:simpleType name="SoftwareVersionType">
        <xsd:annotation>
            <xsd:documentation>
                A string identifying the software version. To allow version comparisons, this element should be in the form of dot-delimited integers, where each successive integer represents a more minor category of variation.  For example, 3.0.21 where the components mean: Major.Minor.Build.
</xsd:documentation>
        </xsd:annotation>
        <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
    <xsd:simpleType name="SerialNumberType">
        <xsd:annotation>
            <xsd:documentation>
                Serial number of the CPE.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
    <xsd:simpleType name="PreferenceType">
        <xsd:annotation>
            <xsd:documentation>
                Preference type is used to order lists in reverse order: the lowest value has the highest preference.
```

```
            </xsd:documentation>
        </xsd:annotation>
        <xsd:restriction base="xsd:positiveInteger"/>
    </xsd:simpleType>
    <xsd:complexType name="InterfaceTypeList">
        <xsd:annotation>
            <xsd:documentation>
                Type to define the list of supported interfaces from the specification document, note that this does not specify the
                protocol used by any of these devices
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence maxOccurs="unbounded">
            <xsd:element name="Interface">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:string">
                        <xsd:enumeration value="'1 Firmware Package'"/>
                        <xsd:enumeration value="'2 Metadata'"/>
                        <xsd:enumeration value="'3 RMS Administrator'"/>
                        <xsd:enumeration value="'4 FUS Interface'"/>
                        <xsd:enumeration value="'5 Multicast Delivery'"/>
                        <xsd:enumeration value="'6 Unicast Delivery'"/>
                        <xsd:enumeration value="'7 Firmware Announcement'"/>
                        <xsd:enumeration value="'8 Query Response Channel'"/>
                        <xsd:enumeration value="'9 CPE Management'"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="DeviceClassType">
        <xsd:annotation>
            <xsd:documentation>
                Structured type to identify device grouping based on the product class of a particular CE manufacturer .
            </xsd:documentation>
        </xsd:annotation>
        <xsd:all>
            <xsd:element name="ManufacturerOUI" type="rms:ManufacturerOUIType"/>
            <xsd:element name="ProductClass" type="rms:ProductClassType"/>
        </xsd:all>
    </xsd:complexType>
    <xsd:complexType name="DeviceClassHardwareVersionType">
        <xsd:annotation>
            <xsd:documentation>
                Stuctured type to identify a class of devices having a particular hardware version.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:all>
            <xsd:element name="DeviceClass" type="rms:DeviceClassType"/>
            <xsd:element name="HardwareVersion" type="rms:HardwareVersionType"/>
        </xsd:all>
    </xsd:complexType>
    <xsd:complexType name="DeviceClassSoftwareVersionType">
        <xsd:annotation>
            <xsd:documentation>
                Structured type to identify a class of devices having a particular software version.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:all>
            <xsd:element name="DeviceClass" type="rms:DeviceClassType"/>
            <xsd:element name="SoftwareVersion" type="rms:SoftwareVersionType"/>
        </xsd:all>
    </xsd:complexType>
    <xsd:complexType name="RangeListType">
        <xsd:annotation>
            <xsd:documentation>
                Type to define the very flexible mode to specify which CPEs need to be upgraded with the new SoftwareVersion
                provided.
            </xsd:documentation>
        </xsd:annotation>
```

```
    <xsd:sequence maxOccurs="unbounded">
        <xsd:choice>
            <xsd:annotation>
                <xsd:documentation>
                    Elements in range-list type can be either a range or a list of multiple specific items.
                </xsd:documentation>
            </xsd:annotation>
            <xsd:element name="Range" maxOccurs="unbounded">
                <xsd:annotation>
                    <xsd:documentation>
                        Generic range as a pair of delimiting values. Used to define contigiuous lists.
                    </xsd:documentation>
                </xsd:annotation>
                <xsd:complexType>
                    <xsd:all>
                        <xsd:element name="LowerBound" type="xsd:string"/>
                        <xsd:element name="UpperBound" type="xsd:string"/>
                    </xsd:all>
                    <xsd:attribute name="Availability" type="xsd:boolean" use="optional">
                        <xsd:annotation>
                            <xsd:documentation>
                                Availability attribute may be used to specify whether the range is valid or to be excluded. "True"
                                indicates "Include" and "False" indicates "Exclude". The default behaviour is "True" when the
                                attribute is not specified.
                            </xsd:documentation>
                        </xsd:annotation>
                    </xsd:attribute>
                </xsd:complexType>
            </xsd:element>

            <xsd:element name="Item" maxOccurs="unbounded">
                <xsd:annotation>
                    <xsd:documentation>
                        Single value for the specific item. The semantic is vendor specific but can be used to create a list of non-
                        contiguous item descriptions.
                    </xsd:documentation>
                </xsd:annotation>
                <xsd:complexType  >
                    <xsd:all>
                        <xsd:element name="ItemID" type="xsd:string"/>
                    </xsd:all>
                <xsd:attribute name="Availability" type="xsd:boolean" use="optional">
                    <xsd:annotation>
                        <xsd:documentation>
                          Availability attribute may be used to specify whether the range is valid or to be excluded. True (1)
                          stands for Include and False (0) stands for Exclude. The default behaviour is True=Include when the
                          attribute is not specified.
                        </xsd:documentation>
                    </xsd:annotation>
                </xsd:attribute>
            </xsd:complexType>
            </xsd:element>
        </xsd:choice>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="DeviceClassInfoType">
    <xsd:annotation>
        <xsd:documentation>
            This type contains device class specific information based on general description components
        </xsd:documentation>
    </xsd:annotation>
    <xsd:all>
        <xsd:element name="ManufacturerOUI" type="rms:ManufacturerOUIType"/>
        <xsd:element name="ProductClass" type="rms:ProductClassType"/>
        <xsd:element name="HardwareVersion" type="rms:HardwareVersionType"/>
        <xsd:element name="SoftwareVersion" type="rms:SoftwareVersionType"/>
    </xsd:all>
</xsd:complexType>
<xsd:complexType name="ResourceAccessInfoType">
```

```
        <xsd:annotation>
            <xsd:documentation>
                Gather all information needed to access a network resource.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
            <xsd:element name="URL" type="xsd:anyURI">
                <xsd:annotation>
                    <xsd:documentation>
                        URL as defined in [RFC 3986]. Where 'dvb-mcast://' format used this may contain the payload type and
                        associated parameters needed to fully specify the specific usage.
                    </xsd:documentation>
                </xsd:annotation>
            </xsd:element>
            <xsd:element name="Username" type="xsd:string" minOccurs="0">
                <xsd:annotation>
                    <xsd:documentation>
                        Optional username used to authenticate the users of the resource when making a connection to the URL. The
                        usage of this parameter depends on the protocol specified in the URL.
                    </xsd:documentation>
                </xsd:annotation>
            </xsd:element>
            <xsd:element name="Password" type="xsd:string" minOccurs="0">
                <xsd:annotation>
                    <xsd:documentation>
                        Optional password used to authenticate the users of the resource when making a connection to the URL. The
                        usage of this parameter depends on the protocol specified in the URL.
                    </xsd:documentation>
                </xsd:annotation>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute name="Protocol" use="optional">
            <xsd:annotation>
                <xsd:documentation>
                    Optional protocol used for payload by URI
                </xsd:documentation>
            </xsd:annotation>
            <xsd:simpleType>
                <xsd:restriction base="xsd:string">
                    <xsd:enumeration value="1 SAP"/>
                    <xsd:enumeration value="2 DVBSTP"/>
                    <xsd:enumeration value="3 FLUTE"/>
                    <xsd:enumeration value="4 MP2T"/>
                    <xsd:enumeration value="5 TCP/IP"/>
                    <xsd:enumeration value="6 UDP/IP"/>
                    <xsd:enumeration value="7 FTP"/>
                    <xsd:enumeration value="8 SFTP"/>
                    <xsd:enumeration value="9 TFTP"/>
                    <xsd:enumeration value="10 IGMP"/>
                    <xsd:enumeration value="11 HTTP"/>
                    <xsd:enumeration value="12 HTTPS"/>
                    <xsd:enumeration value="13 SOAP"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:attribute>
    </xsd:complexType>
<!-- End of TypeDefinitions -->

<!-- RMS-FUS Schema definition -->
    <xsd:element name="RMS-FUS_announcement" >
        <xsd:annotation>
            <xsd:documentation>
                Metadata schema definition for DVB TM-IPI RMS-FUS Phase 2 purposes
            </xsd:documentation>
        </xsd:annotation>
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="Mode" minOccurs="0">
```

```
        <xsd:annotation>
            <xsd:documentation>
                Explicit declaration as to whether firmware update operation is managed or unmanaged.  Mode = "RMS"
                for managed, mode = "FUS" for unmanaged
            </xsd:documentation>
        </xsd:annotation>
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="RMS"/>
                <xsd:enumeration value="FUS"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element name="EntityDefinition">
        <xsd:annotation>
            <xsd:documentation>
                Schema structure is split into sections to include information for CEManufacturer , FUS, RMS,
                TargetDevice, FirmwareUpgrade and Message.  The XML document based on this schema may not be
                fully populated
            </xsd:documentation>
        </xsd:annotation>
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="CEManufacturerInfo" minOccurs="0">
                    <xsd:annotation>
                        <xsd:documentation>
                            Container for the CE manufacturer information.
                        </xsd:documentation>
                    </xsd:annotation>
                    <xsd:complexType>
                        <xsd:sequence>
                            <xsd:element name="Manufacturer"  type="xsd:string" minOccurs="0">
                                <xsd:annotation>
                                    <xsd:documentation>
                                        The manufacturer of the CPE (human readable string).
                                    </xsd:documentation>
                                                                        </xsd:annotation>
                                                                    </xsd:element>
                                                                    <xsd:element name="ManufacturerOUI"
                                    type="rms:ManufacturerOUIType">
                                                                        <xsd:annotation>
                                                                            <xsd:documentation>OUI of the
                                    CE manufacturer.</xsd:documentation>
                                                                        </xsd:annotation>
                                                                    </xsd:element>
                                                                    <xsd:element name="FirmwareLocation"
                                    type="rms:ResourceAccessInfoType" minOccurs="0">
                                <xsd:annotation>
                                    <xsd:documentation>
                                        Information concerning the location of the deployed firmware upgrade package
                                        within the CE manufacturer repository.
                                    </xsd:documentation>
                                </xsd:annotation>
                            </xsd:element>
                            <xsd:element name="ReportingOrders" minOccurs="0">
                                <xsd:annotation>
                                    <xsd:documentation>
                                        Grouping of information about option for Manufacturer to request report from CPE
                                        about firmware update
                                    </xsd:documentation>
                                </xsd:annotation>
                                <xsd:complexType>
                                    <xsd:sequence>
                                        <xsd:element name="ReportingInterfaceLocation" maxOccurs="unbounded">
                                            <xsd:annotation>
                                                <xsd:documentation>
                                                    The address to which a CPE should report after a firmware update,
                                                    this is intended for use by unmanaged CPEs to report to the CE
                                                    manufacturer, possibly via a 3rd party.
```

```
                                        </xsd:documentation>
                                    </xsd:annotation>
                                    <xsd:complexType>
                                        <xsd:complexContent>
                                            <xsd:extension base="rms:ResourceAccessInfoType">
                                    <xsd:attribute name="Preference"
type="rms:PreferenceType"use="required"/>
                                            </xsd:extension>
                                        </xsd:complexContent>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="VendorSpecificInfo" type="xsd:anyType" minOccurs="0">
                                    <xsd:annotation>
                                        <xsd:documentation>
                                            Other unspecified information specific to a firmware update for
                                            managed or unmanaged CPEs.  The format is not specified.
                                        </xsd:documentation>
                                    </xsd:annotation>
                                </xsd:element>
                            </xsd:sequence>
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="FUSInfo" minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>
                    Container for the FUS information.  Tables 4 and 5 in TS 102 824 v1.2.1 contains the
                    requirements for the interfaces on the FUS and the CPE.
                </xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="Name" type="xsd:string" minOccurs="0">
                        <xsd:annotation>
                            <xsd:documentation>
                                Optional name of the FUS in a human readable string format.
                            </xsd:documentation>
                        </xsd:annotation>
                    </xsd:element>
                    <xsd:element name="ID" type="xsd:string" minOccurs="0">
                        <xsd:annotation>
                            <xsd:documentation>Optional identifier of the FUS.</xsd:documentation>
                        </xsd:annotation>
                    </xsd:element>
                    <xsd:element name="SourceAddress" type="xsd:string" minOccurs="0">
                        <xsd:annotation>
                            <xsd:documentation>Source address of the FUS.</xsd:documentation>
                        </xsd:annotation>
                    </xsd:element>
                    <xsd:element name="FUSQueryInterfaceLocation" minOccurs="0"
maxOccurs="unbounded">
                        <xsd:annotation>
                            <xsd:documentation>
                                Unicast Query Interface.
                            </xsd:documentation>
                        </xsd:annotation>
                        <xsd:complexType>
                            <xsd:complexContent>
                                <xsd:extension base="rms:ResourceAccessInfoType">
                                <xsd:attribute name="Preference" type="rms:PreferenceType" use="required"/>
                                </xsd:extension>
                            </xsd:complexContent>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="MulticastInterfaceLocation" minOccurs="0"
maxOccurs="unbounded">
```

```
            <xsd:annotation>
                <xsd:documentation>
                    Multicast firmware delivery interface.
                </xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
                <xsd:complexContent>
                    <xsd:extension base="rms:ResourceAccessInfoType">
                    <xsd:attribute name="Preference" type="rms:PreferenceType" use="required"/>
                    </xsd:extension>
                </xsd:complexContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="MulticastAnnouncementInterfaceLocation" maxOccurs="unbounded">
            <xsd:annotation>
                <xsd:documentation>
                    Multicast announcement interface.
                </xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
                <xsd:complexContent>
                    <xsd:extension base="rms:ResourceAccessInfoType">
                    <xsd:attribute name="Preference" type="rms:PreferenceType" use="required"/>
                        <xsd:attribute name="ScanLocation" type="xsd:boolean" use="optional">
                            <xsd:annotation>
                                <xsd:documentation>
                                    ScanLocation attribute may be used to indicate that the
                                    announcement URL is a pointer message.  "True" indicates that it
                                    is a pointer, default is "false" indicating a description message.
                                </xsd:documentation>
                            </xsd:annotation>
                        </xsd:attribute>
                    </xsd:extension>
                </xsd:complexContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="UnicastInterfaceLocation" maxOccurs="unbounded">
            <xsd:annotation>
                <xsd:documentation>
                    Unicast firmware update delivery interface.
                </xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
                <xsd:complexContent>
                    <xsd:extension base="rms:ResourceAccessInfoType">
                        <xsd:attribute name="Preference" type="rms:PreferenceType"
use="required"/>
                    </xsd:extension>
                </xsd:complexContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="SupportedInterfaces" type="rms:InterfaceTypeList"  minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>
                    A combination of interfaces 5 to 8 as specified in tables 4 and 5 of TS 102 824
                    v1.2.1 shall be defined for the FUS.
                </xsd:documentation>
            </xsd:annotation>
        </xsd:element>
        <xsd:element name="DeliveryRate" minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>
                    Maximum delivery rate capability for FUS in bps.
                </xsd:documentation>
            </xsd:annotation>
            <xsd:simpleType>
                <xsd:restriction base="xsd:positiveInteger"/>
            </xsd:simpleType>
        </xsd:element>
```

```
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="RMSInfo" minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>Container for the RMS information.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="Name" type="xsd:string" minOccurs="0">
                        <xsd:annotation>
                            <xsd:documentation>
                                Optional name of the RMS: human readable string.
                            </xsd:documentation>
                        </xsd:annotation>
                    </xsd:element>
                    <xsd:element name="ID" type="xsd:string" minOccurs="0">
                        <xsd:annotation>
                            <xsd:documentation>Optional identifier of the RMS.</xsd:documentation>
                        </xsd:annotation>
                    </xsd:element>
                    <xsd:element name="ManagementServerLocation" type="rms:ResourceAccessInfoType">
                        <xsd:annotation>
                            <xsd:documentation>
                                Management server interface, this is mandatory for the RMS if it is used.
                            </xsd:documentation>
                            </xsd:annotation>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="TargetDeviceInfo" minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>
                    Information container of the target devices for the  software upgrade currently described in
                    the announcement message (XML  description document or pointer).
                </xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="SupportedInterfaces" type="rms:InterfaceTypeList">
                        <xsd:annotation>
                            <xsd:documentation>
                                The combination of interfaces 5 to 9 which are necessary on the target CPEs for
                                delivery of this firmware update.
                            </xsd:documentation>
                        </xsd:annotation>
                    </xsd:element>
                    <xsd:element name="VendorSpecificInfo" type="xsd:anyType" minOccurs="0">
                        <xsd:annotation>
                            <xsd:documentation>
                                Vendor specific information container used to provide specific inforrmation about
                                the interfaces which are necessary on the target CPEs for this firmware update.
                            </xsd:documentation>
                        </xsd:annotation>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<xsd:element name="FirmwareUpgradeInfo">
    <xsd:annotation>
        <xsd:documentation>
            Contains information about the software package to be uploaded. Any XML instance of this schema
            describes a single software package deployed.
        </xsd:documentation>
    </xsd:annotation>
```

```
<xsd:complexType>
    <xsd:sequence>
        <xsd:element name="DeviceClassSoftwareVersion" type="rms:DeviceClassSoftwareVersionType">
            <xsd:annotation>
                <xsd:documentation>
                    Identify the target CPEs by current software version and Device Class.
                </xsd:documentation>
            </xsd:annotation>
        </xsd:element>
        <xsd:element name="TargetDevices">
            <xsd:annotation>
                <xsd:documentation>
                    Devices to be upgraded can be identified by Device Class or based on a combination of
                    device filtering using a more sophisticated means including use of serial numbers and ranges.
                </xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
                <xsd:choice>
                    <xsd:element name="DeviceClassInfo" type="rms:DeviceClassInfoType">
                        <xsd:annotation>
                            <xsd:documentation>
                                This element acts as a filter for a single class of devices.
                            </xsd:documentation>
                        </xsd:annotation>
                    </xsd:element>
                    <xsd:element name="DeviceGroup">
                        <xsd:annotation>
                            <xsd:documentation>
                                This element acts as a very flexible filter to identify the set of devices to be
                                uploaded.
                            </xsd:documentation>
                        </xsd:annotation>
                        <xsd:complexType>
                            <xsd:all>
                                <xsd:annotation>
                                    <xsd:documentation>
                                        All the contained elements have to be considered as AND-ed to identify
                                        the target devices.
                                    </xsd:documentation>
                                </xsd:annotation>
                                <xsd:element name="DeviceClass" type="rms:DeviceClassType">
                                    <xsd:annotation>
                                        <xsd:documentation>
                                            Target devices are from a class of devices.
                                        </xsd:documentation>
                                    </xsd:annotation>
                                </xsd:element>
                                <xsd:element name="HardwareVersionList" minOccurs="0">
                                    <xsd:complexType>
                                        <xsd:sequence maxOccurs="unbounded">
                                            <xsd:annotation>
                                                <xsd:documentation>
                                                    Hardware versions in this list shall be OR-ed to identify the
                                                    target devices.
                                                </xsd:documentation>
                                            </xsd:annotation>
                                            <xsd:element name="HardwareVersion"
type="rms:HardwareVersionType"/>
                                        </xsd:sequence>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="SoftwareVersionList" minOccurs="0">
                                    <xsd:complexType>
                                        <xsd:sequence maxOccurs="unbounded">
                                            <xsd:annotation>
                                                <xsd:documentation>
                                                    Software versions in this list shall be OR-ed to identify the
                                                    target devices
```

```
                                        </xsd:documentation>
                                    </xsd:annotation>
                                    <xsd:element name="SoftwareVersion"
type="rms:SoftwareVersionType"/>
                                </xsd:sequence>
                            </xsd:complexType>
                        </xsd:element>
                        <xsd:element name="SerialNumberRangeList" type="rms:RangeListType"
minOccurs="0"/>
                        <xsd:element name="MACAddressRangeList" type="rms:RangeListType"
minOccurs="0"/>
                        <xsd:element name="VendorSpecificInfo" type="xsd:anyType" minOccurs="0">
                            <xsd:annotation>
                                <xsd:documentation>
                                    Vendor specific information container, the details of this are out of
                                    scope.
                                </xsd:documentation>
                            </xsd:annotation>
                        </xsd:element>
                    </xsd:all>
                </xsd:complexType>
            </xsd:element>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="ValidityTimeRange" minOccurs="0">
    <xsd:complexType>
        <xsd:all>
            <xsd:element name="StartTime" type="xsd:dateTime"/>
            <xsd:element name="EndTime" type="xsd:dateTime"/>
        </xsd:all>
    </xsd:complexType>
</xsd:element>
<xsd:element name="SoftwarePackageInfo" minOccurs="0">
    <xsd:annotation>
        <xsd:documentation>
            Optional information of the software package. If the software upgrade is composed by a
            single file or package the PackageName and PackageSize shall be used. PackageFile element
            shall be used only when more than a single file are part of the same software package.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
        <xsd:sequence maxOccurs="unbounded">
            <xsd:element name="PackageName" type="xsd:string" minOccurs="0">
                <xsd:annotation>
                    <xsd:documentation>
                        Opaque string with no specific requirements for its format. The value is to be
                        interpreted based on the CPEs vendor-specific package naming conventions.
                    </xsd:documentation>
                </xsd:annotation>
            </xsd:element>
            <xsd:element name="PackageSize" type="xsd:unsignedLong">
                <xsd:annotation>
                    <xsd:documentation>The size of the package in bytes.</xsd:documentation>
                </xsd:annotation>
            </xsd:element>
            <xsd:element name="PackageFile" minOccurs="0" maxOccurs="unbounded">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="FileName" type="xsd:string">
                            <xsd:annotation>
                                <xsd:documentation>
                                    Opaque string with no specific requirements for its format. The value
                                    is to be interpreted based on the CPEs vendor-specific file naming
                                    conventions.
                                </xsd:documentation>
                            </xsd:annotation>
                        </xsd:element>
                        <xsd:element name="FileSize" type="xsd:unsignedLong" minOccurs="0">
```

```
                        <xsd:annotation>
                            <xsd:documentation>The size of the file in bytes. </xsd:documentation>
                        </xsd:annotation>
                    </xsd:element>
                    <xsd:element name="VendorSpecificInfo" type="xsd:anyType" minOccurs="0">
                        <xsd:annotation>
                            <xsd:documentation>
                                Vendor specific information container, the details of this are out of
scope.
                            </xsd:documentation>
                        </xsd:annotation>
                    </xsd:element>
                </xsd:sequence>
                <xsd:attribute name="ModuleType" type="xsd:string">
                    <xsd:annotation>
                        <xsd:documentation>
                            Attribute list for the module type to be defined.
                        </xsd:documentation>
                    </xsd:annotation>
                </xsd:attribute>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="VendorSpecificInfo" type="xsd:anyType" minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>
                    Vendor specific information container, the details of this are out of scope.
                </xsd:documentation>
            </xsd:annotation>
        </xsd:element>
        <xsd:element name="FootprintSize" minOccurs="0" maxOccurs="unbounded">
            <xsd:complexType>
                <xsd:annotation>
                    <xsd:documentation>
                        Required available size of installed image - parent element
                    </xsd:documentation>
                </xsd:annotation>
                <xsd:attribute name="Volatile" type="xsd:unsignedLong">
                    <xsd:annotation>
                        <xsd:documentation>
                            Required available size of installed image in bytes - volatile memory.
                        </xsd:documentation>
                    </xsd:annotation>
                </xsd:attribute>
                <xsd:attribute name="NonVolatile" type="xsd:unsignedLong">
                    <xsd:annotation>
                        <xsd:documentation>
                            Required available size of installed image in bytes - non-volatile memory.
                        </xsd:documentation>
                    </xsd:annotation>
                </xsd:attribute>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ImagePackaging" minOccurs="0">
            <xsd:complexType>
                <xsd:annotation>
                    <xsd:documentation>
                        Switch indicating that the image is packaged and signed - 0 = clear, 1 =
                        packaged and signed.
                    </xsd:documentation>
                </xsd:annotation>
                <xsd:attribute name="SignedPackaging" type="xsd:boolean">
                    <xsd:annotation>
                        <xsd:documentation>
                            Switch indicating that a manifest is used - 0 = false, 1 = true
                        </xsd:documentation>
                    </xsd:annotation>
                </xsd:attribute>
                <xsd:attribute name="Manifest" type="xsd:boolean">
```

```
                        <xsd:annotation>
                            <xsd:documentation>
                                Switch indicating that a manifest is used - 0 = false, 1 = true
                            </xsd:documentation>
                        </xsd:annotation>
                    </xsd:attribute>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="UpdateDescriptor" maxOccurs="1" minOccurs="0">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="UpdateFlag">
                            <xsd:annotation>
                                <xsd:documentation>
                                    Indication of whether update should be forced
                                </xsd:documentation>
                            </xsd:annotation>
                            <xsd:simpleType>
                                <xsd:restriction base="xsd:string">
                                    <xsd:enumeration value="'1 Manual'"/>
                                    <xsd:enumeration value="'2 Automatic'"/>
                                </xsd:restriction>
                            </xsd:simpleType>
                        </xsd:element>
                        <xsd:element name="UpdateMethod">
                            <xsd:annotation>
                                <xsd:documentation>
                                    Indication of when update should be installed
                                </xsd:documentation>
                            </xsd:annotation>
                            <xsd:simpleType>
                                <xsd:restriction base="xsd:string">
                                    <xsd:enumeration value="'1 Immediate'"/>
                                    <xsd:enumeration value="'2 UserConvenience'"/>
                                    <xsd:enumeration value="'3 NextRestart'"/>
                                </xsd:restriction>
                            </xsd:simpleType>
                        </xsd:element>
                        <xsd:element name="UpdatePriority">
                            <xsd:annotation>
                                <xsd:documentation>
                                    Indication of how important update is - 1 - 4, 1 is highest
                                </xsd:documentation>
                            </xsd:annotation>
                            <xsd:simpleType>
                                <xsd:restriction base="xsd:integer"/>
                            </xsd:simpleType>
                        </xsd:element>
                    </xsd:sequence>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
<xsd:element name="MessageDescriptor" type="xsd:string" minOccurs="0">
    <xsd:annotation>
        <xsd:documentation>
```

```
                                The message may be used to inform the user about the purpose of this System Software
                                Update (SSU) in order to receive the users consent to perform the actual update.
                            </xsd:documentation>
                        </xsd:annotation>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:schema>
```

Annex C (normative): Uniform Resource Identifier (URI) scheme for the connection to a multicast service

A URI may be used to provide a means to locate the multicast group carrying the announcement service or firmware update and also to specify information concerning the applicative layer transport protocol which will be used to carry the data stream over that multicast channel (e.g. FLUTE protocol, as defined in [20]). For each transport protocol some additional parameters may optionally be provided to give more accurate redirection as described in the sections below.

The Internet Group Management Protocol (IGMP) is primarily considered as the appropriate connection protocol to be used when a network entity has to join a multicast group [19], either in Any Source Multicast (ASM) mode using IGMPv2 [29] or IGMPv3 [30], or in Source Specific Multicast (SSM) mode using IGMPv3 [30], and the "dvb-mcast" URI described in clause G.3 of the IPTV Handbook, TS 102 034 v1.4 [1] is specified to carry all the required parameters to support such a connection.

Further profiling of the dvb-mcast URI specifically required for use in this RMS-FUS application is described below, e.g. CarouselId for DSM-CC.

C.1 Basic DVB-MCAST URI scheme

The basic DVB-MCAST URI schema as defined in clause G.3.1 of TS 102 034 [1] is re-used in the present document, it is duplicated for information here. Additional Payload types appropriate to RMS-FUS usage are included relative to the definition in TS 102 034 [1]:

```
'dvb-mcast://' [ src-host '@' ] mcast-addr ':' port ['?payload=' payloadID]

src-host        = source host
mcast-addr      = multicast address
port           = port
payloadID       = payload-type
payload-type   = "sap" | "dvbstp" | "flute" | "mp2t"
```

The inclusion of the multicast address, port and payload is mandatory in the usage for RMS-FUS, the source host is optional but required in cases where SSM mode is used.

C.2 RMS-FUS usage of DVB-MCAST URI scheme for DVBSTP

The DVB-MCAST URI scheme defined in clause G.3.2 of TS 102 034 [1] is re-used in the present document. However, the CDS specific parameter carrying the ‘Download-Session-ID' is not relevant in this case.

The resulting DVB-MCAST URI scheme for DVBSTP for RMS-FUS is defined as follows:

```
'dvb-mcast://' [ src-host '@' ] mcast-addr ':' port '?payload=dvbstp' ['&service-provider='
ServiceProviderID] ['&dvbstp-payload=' DVBSTPPayloadID] ['&segment=' SegmentID]

ServiceProviderID  = IPv4 address
DVBSTPPayloadId  = 2*2 HEXDIG, this is set to 0xB2 for RMS-FUS
SegmentId          = 4*4 HEXDIG; hex number in range 0x0000 to 0xffff
```

If present the value of "DVBSTPPayloadId" for RMS-FUS announcement messages is defined to be 0xB2, the additional SegmentID parameter may optionally point to the specific segment of the XML document.

C.3 RMS-FUS usage of DVB-MCAST URI scheme for SAP

The basic DVB-MCAST URI scheme for SAP is defined in clause G.3.3 of TS 102 034 [1] and is re-used in the present document.

In order to reference a specific session description within the SDP information the RMS-FUS SDP session_ID can be provided within the fragment part of the URI. The fragment syntax is not specific to the SAP delivery but to the delivered media, the SDP session description in this case.

The resulting DVB-MCAST URI scheme for SAP for RMS-FUS is defined as follows:

```
'dvb-mcast://' [ src-host '@' ] mcast-addr ':' port '?payload=sap' ['#? sdp-session-id='RMS-
FUS_session_ID]

RMS-FUS_session_ID   = unsigned Integer
```

The "RMS-FUS_session_ID" shall be the same as that used to identify the SDP session in the header parameter ("o=") as described in table 10.

If a RMS-FUS_session_ID is provided in the fragment component of the URI, the device has to search all the SDP information delivered over the multicast channel for the session description with the specific RMS-FUS_session_ID.

C.4 RMS-FUS usage of DVB-MCAST URI scheme for DSM-CC

TS 102 851 [38] defines the DVB URI which allows differentiation between services and components transported in the MPEG-2 transport stream. For the DVB-MCAST usage the Orignal Network ID and Transport Strem ID are replaced by the IPTV mulicast channel address informtion (multicast address, port and optional source address).

The DVB-MCAST payload parameter is set to 'mp2t' to indicate a MPEG-2 transport stream directly transported via UDP as this is the only mapping supported for RMS/FUS. Note that for a MPEG-2 transport stream transported via RTP/UDP the parameter 'mp2t/rtp' shall be used.

For referencing a DSM-CC carousel with firmware update files only a subset of this URI format is used. If only the multicast address information (source host, multicast address and port) is provided the whole MPEG-2 transport stream is referenced. The CPE has to look on all DSM-CC carousels in the transport stream in order to locate the corect firmware download. With the addition of the Service ID a specific elementary stream within the transport stream can be referenced. With the addition of the Carousel ID a specific DSM-CC caroousel within the transport stream can be referenced, such avoiding that the CPE has to look into all carousels.

The resulting DVB-MCAST URI schema for RMS-FUS DSM-CC usage is defined as follows:

```
'dvb-mcast://' [ src-host '@' ] mcast-addr ':' port '?payload=mp2t' ['&dvb-service='  service_id
["." component_set [$<dvb_carousel_id>]]]

service_id      = as defined in TS 102 851 [38], clause 6.1
component_set   = as defined in TS 102 851 [38], clause 6.1
dvb_carousel_id = as defined in TS 102 851 [38], clause 6.1
```

C.5 RMS-FUS usage of DVB-MCAST URI scheme for FLUTE

The basic DVB-MCAST URI scheme defined in clause G.3.1 of TS 102 034 [1] is extended in the present document in order to reference FLUTE sessions and individual files within a FLUTE session. The FLUTE TSI and File URI are defined as part of the Query component of the URI as they provide non-hierarchical information to locate a specific FLUTE session and file.

This DVB-MCAST URI for RMS-FUS FLUTE usage covers only the case of a FLUTE session using a single multicast channel and without FEC. Multiple multicast channels or FEC for a FLUTE session require additional parameters (e.g. multicast address and port per multicast channel) which are not covered by this scheme.

The resulting DVB-MCAST URI scheme for FLUTE for RMS-FUS is defined as follows:

```
'dvb-mcast://' [ src-host '@' ] mcast-addr ':' port '?payload=flute' [&tsi=TSI] ['file-uri='
File_URI]

tsi      = TSI of Flute session
file_URI = File URI as used for the content-location attribute of the FLUTE FDT
```

Multiple files (objects) may be carried in a single FLUTE session, but any single file must be contained entirely in that session. To identify which file shall be downloaded the File_URI as used in the content-location attribute of the FLUTE FDT shall is provided. All file within the session will be downloaded if no no file_URI is provided.

Multiple FLUTE sessions may be carried on a single multicast channel in which case they are identified by the TSI.

Annex D (normative): SDP attributes specifically defined for DVB RMS

In order to be able to supply the appropriate information to the CPEs in an efficient way a number of DVB RMS specific attributes must be defined. Such additional attribute names, except for FLUTE specific attributes which are defined in [37], start with "x-" to indicate that they have not been adopted by IETF, the "dvb-rms-" then indicates that the use of the attributes is within the scope of the present document.

A single SDP announcement message may contain more than one media-description section.

Each media-description section represents an announcement message for a target population of CPE identified by the associated target filtering attributes described below.

Each media-description starts with an "m=" field and is terminated by either the next "m=" field or by the end of the SDP message.

The media description carries the information about the data source (i.e. network address) and the announcement type (i.e. the protocol which must be used when listening from the data source specified).

Each media-description connection information shall be followed by one or more attribute lines ("a=" fields) that are media session specific and add information about the media stream. Attribute lines may be used to specify, for example:

- the source IP address of the multicast stream, in case a multicast stream is announced;
- the TSI (Transport Session Identifier) in cases where FLUTE is announced;
- the target devices that each media-description refers to.

The media description line "m=" is mandatory and contains several sub-fields with the following syntax, as specified in clause 5.14 of [26]:

```
m=application <port>/<number of ports> <proto> <fmt> ...
```

The <proto> values in the table D.1 are defined for DVB purposes, please refer to http://www.iana.org/assignments/sdp-parameters for the list of registered parameters.

Table D.1: DVB media protocols for application media

Type	<proto> value	<fmt> value	Description
Pointer / announcement	SAP/UDP	SDP	The media description refers to another multicast announcement stream. The <port> is the UDP/IP destination port used to deliver the data stream.
	DVBSTP/UDP	XML	
Multicast update	FLUTE/UDP	*	The media description refers to a multicast FLUTE or DSM-CC (MPEG-2 transport stream) session that delivers the firmware update files. The <port> is the UDP/IP destination port used by the multicast session to deliver the data stream. The "*" shall be used for the <fmt> value indicating that miscellaneous and unspecified MIME types (file formats) are contained in the session.
	MP2T/UDP	*	
Query	HTTP/TCP	SOAP	The media description is used to command the target CPEs to query the FUS via interface 8 (QRC). The <port> is the one the client should use when connecting to the QRC interface.
Unicast update	FTP/TCP	*	The media description refers to a unicast location for the firmware update files. The <port> is the one the client should use when connecting to the unicast interface by using the specified transport protocol. The "*" shall be used for the <fmt> value indicating that miscellaneous and unspecified MIME types (file formats) are contained in the session.
	HTTP/TCP	*	
	SFTP/TCP	*	
	HTTPS/TCP	*	
	TFTP/TCP	*	

The connection information line "c=" is mandatory and contains several sub-fields with the following syntax, as specified in clause 5.7 of [26]:

```
c=IN IP4 <connection-address>
```

The connection information must be used according to the standard SDP specification, RFC 4566 [26].

- If the announced session is multicast (pointer announcement or update announcement) then the connection address will be an IP multicast group address.
- If the announced session is unicast (query announcement or unicast announcement), then the connection address is provided by a specific URL attribute and the connection-address of the "c=" line is set to "0.0.0.0".

D.1 FLUTE specific attributes (multicast update announcement)

FLUTE specific SDP attributes are defined in [37], clause 6.1.13. Attributes that are necessary in the RMS context are:

Transport Session Idendifier (TSI) for the FLUTE session:

```
a= flute-tsi:<tsi>
```

Source filter defining the source address in case of Source Specific Multicast (SSM) [36]:

```
a= source-filter: incl IN addr-type * <src-unicast-address>
```

The multicast address of the FLUTE session is provided by the connection data field "c=". The port number is provided by the port sub-field of the media announcement field "m=" with a m-line for each channel.

EXAMPLE:

Message including SDP and media coding for the FLUTE announcement:

```
v=0
o=dvb-fus 2890844526 2890842807 IN IP4 10.47.16.5
s=dvb-update
t=2873397496 2873404696
a=x-dvb-rms-ce-manufacturer:00D09D
m=application 49153 FLUTE/UDP *
c=IN IP4 240.0.0.3
a=flute-tsi:142
a=source-filter: incl IN IP4 * 192.168.1.1
....
```

In this example the media-description is used to announce a single channel FLUTE session delivery to the destination port 49153 and multicast IPv4 address 240.0.0.3. The FLUTE data stream will be sent from the address 192.168.1.1 and is identified by the TSI=142.

Additionally, some optional attributes may be used.

Session timing parameters (start and end time) may be provided by the timing field "t=".

Bandwidth parameters by using SDP bandwidth modifiers as specified in [26].

Timeout values for the fragment wait, table wait and object wait timers using:

```
"a=session-timeout:<fragement>; <table>; <object>"
```

Number of channels may be indicated using "`a= flute-channel:<number of channels>`", in the absence of this attribute a single FLUTE channel is assumed. If multiple FLUTE channels are used each channel will have its associated "m=" line and only a single population of CPEs can be supported by this announcement.

To identify a specific file within the FLUTE session:

"a=x-dvb-rms-file-uri: <File URI>"

File URI is the URI for the specific file in the content-location attribute of the FLUTE FDT.

D.2 DSM-CC specific attributes (multicast update announcement)

The multicast address of the MPEG-2 transport stream with the DSM-CC carousel is provided by the connection data field "c=". The port number is provided by the port sub-field of the media announcement field "m=".

Source filter defining the source address in case of Source Specific Multicast (SSM) [36]:

```
a= source-filter: incl IN addr-type * <src-unicast-address>
```

Session timing parameters (start and end time) may be provided by the timing field "t=".

Bandwidth parameters may be provided using the SDP bandwidth modifiers ("b=") field as specified in [26].

EXAMPLE:

Message including SDP and media coding for the DSM-CC announcement:

```
v=0
o=dvb-fus 2890844526 2890842807 IN IP4 10.47.16.5
s=dvb-update
t=2873397496 2873404696
a=x-dvb-rms-ce-manufacturer: 00D09D
m=application 49153 MP2T/UDP *
c=IN IP4 240.0.0.3
a=source-filter: incl IN IP4 * 192.168.1.1
….
```

In this example the media description is used to announce a MPEG-2 transport stream delivery to the destination port 49153 and multicast IPv4 address 240.0.0.3. The FLUTE data stream will be sent from the address 192.168.1.1 and is identified by the TSI=142.

Additionally, some optional attributes may be used.

The MPEG-2 transport stream service ID to identify a specific service within the transport stream:

"a=x-dvb-serivce-id=<service_id>"

service_id = as defined in TS 102 851 [38], clause 6.1

The carousel ID to identify a specific carousel in the transport stream:

"a=x-dvb-carousel-id=<dvb_carousel_id>"

dvb_carousel_id = as defined in TS 102 851 [38], clause 6.1

D.3 SAP media type usage (pointer announcement)

A pointer announcement to a another SDP announcement message transported via SAP must be specified, with the following syntax:

```
m=application <port> SAP/UDP SDP
```

The multicast address is provided by the connection data field "c=". The port number is provided by the port sub-field of the media announcement field "m=".

Source filter defining the source address in case of Source Specific Multicast (SSM) [36]:

a= source-filter: incl IN addr-type * <src-unicast-address>

In order to reference a specific session description within the SDP information the RMS-FUS SDP session_ID can be provided:

"a=x-dvb-sdp-session: <SDP_session_ID>"

SDP_session_ID = unsigned Integer

The "SDP_session_ID" shall be the same as that used to identify the SDP/SAP session in the header parameter ("o=") as described in table 10.

EXAMPLE:

Message including SDP and media coding for the pointer announcement:

```
v=0
o=dvb-fus 2890844526 2890842807 IN IP4 10.47.16.5
s=dvb-update
t=2873397496 2873404696
a=x-dvb-rms-ce-manufacturer:00D09D
m=application 49155 SAP/UDP SDP
c=IN IP4 240.0.0.3
a=source-filter:IN IP4 incl 192.168.1.1
….
```

In this example the media-description is used to announce a pointer to another announcement delivered to the destination port 49153 and multicast IPv4 address 240.0.0.3. The announcement will be sent from the address 192.168.1.1. The example does not include target filtering apart from the session-level attribute "`a=x-dvb-rms-ce-manufacturer`".

D.4 DVBSTP media type usage (pointer announcement)

A pointer announcement to a XML announcement message, transported via DVBSTP must be specified, with the following syntax:

m=application <port> DVBSTP/UDP XML

The multicast address is provided by the connection data field "c=". The port number is provided by the port sub-field of the media announcement field "m=".

Source filter defining the source address in case of Source Specific Multicast (SSM) [36]:

a= source-filter: incl IN addr-type * <src-unicast-address>

DVBSTP payload ID, service provider ID and segment ID can be specified as additional parameters:

"a=x-dvb-dvbstp-payload: <DVBSTPPayloadID>"

DVBSTPPayloadID = 2*2 HEXDIG

"a=x-dvb-dvbstp-service-provider: <ServiceProviderID>"

ServiceProviderID = IPv4 address

"a=x-dvb-dvbstp-segment: <SegmentID>"

SegmentId = 4*4 HEXDIG; any hex number from 0x0000 to 0xffff

If present the value of "DVBSTPPayloadId" for RMS-FUS announcement messages is defined to be 0xB2, the additional SegmentID parameter may optionally point to the specific segment of the XML document.

EXAMPLE:

Message including SDP and media coding for the XML/DVBSTP pointer announcement:

```
v=0
o=dvb-fus 2890844526 2890842807 IN IP4 10.47.16.5
s=dvb-update
t=2873397496 2873404696
a=x-dvb-rms-ce-manufacturer: 00D09D
m=application 49155 DVBSTP/UDP XML
c=IN IP4 240.0.0.3
a=source-filter:IN IP4 incl 192.168.1.1
….
```

In this example the media description is used to announce a pointer to XML announcement delivered to the destination port 49153 and multicast IPv4 address 240.0.0.3. The announcement will be sent from the address 192.168.1.1. The example does not include target filtering apart from the session level attribute "a=x-dvb-rms-ce-manufacturer".

D.5 Query announcement

The media-description section may be used to carry the location of the QRC interface where the target CPE may connect to in order to check whether a new software version is available. The syntax for this media type is:

```
m=application <port> HTTP/TCP SOAP
```

Since the query announcement is used to provide the CPE with a unicast address, the attribute "`a=x-dvb-rms-source-filter`" must not be used.

Furthermore the media description is not suitable to carry information about the URL, therefore the following attribute must be included in the media-description session:

```
a=x-dvb-rms-qrc:<URL>
```

EXAMPLE:

Message including SDP and media coding for the query announcement:

```
v=0
o=dvb-fus 2890844526 2890842807 IN IP4 10.47.16.5
s=dvb-update
t=2873397496 2873404696
a=x-dvb-rms-ce-manufacturer:00D09D
m=application 8080 HTTP/TCP SOAP
c=IN IP4 0.0.0.0
a=x-dvb-rms-qrc:http://soap.example.dvb.rms/QRC
….
```

The server address information is provided by the URL and therefore the address in the "c=" line shall be set to "0.0.0.0" and ignored by the CPE. This allows the use of hostname and path.

In this example the media-description is used to inform the CPEs, which have manufacturerOUI 00D09D, they can query the URL http://soap.example.dvb.rms/QRC at port 8080 to query if there is a new software available by using the SOAP protocol as it is explained in interface 8.

D.6 Unicast announcement

The media-description section may be used to carry the location of the unicast interface (interface 6) where the target CPE may connect to in order to download the new software version. The syntax for this media type is:

m=application <port> FTP/TCP *

m=application <port> HTTP/TCP *

m=application <port> SFTP/TCP *

m=application <port> HTTPS/TCP *

m=application <port> TFTP/TCP *

Since the unicast announcement is used to provide the CPE a unicast address, the attribute "`a=x-dvb-rms-source-filter`" must not be used.

Furthermore the media description is not suitable to carry information about the URL where the download file can be retrieved, therefore the following attribute must be included in the media-description session:

```
a=x-dvb-rms-unicast:<URL>
```

If username and password are needed to access the unicast service they can be provided as part of the URL.

EXAMPLE:

Message including SDP and media coding for the unicast (FTP) announcement:

```
v=0
o=dvb-fus 2890844526 2890842807 IN IP4 10.47.16.5
s=dvb-update
t=2873397496 2873404696
a=x-dvb-rms-ce-manufacturer:00D09D
m=application 21 FTP/TCP *
c=IN IP4 0.0.0.0
a=x-dvb-rms-unicast:ftp://example.dvb.rms/firmware/package
….
```

The server address information is provided by the URL and therefore the address in the "c=" line shall be set to "0.0.0.0" and ignored by the CPE. This allows the use of hostname and path. The CPE shall use the port defined by the "m=" line parameter.

In this example the media-description is used to inform the CPEs, which have manufacturerOUI 00D09D, they can start to download the software named "package" by using the FTP protocol to connect to the server whose address is described in the connection information line.

D.7 Target filtering attributes

Zero or more filtering attributes may be used in the media-description section of the announcement. In cases where there is more than a single attribute line, the filtering conditions represented by each line shall be evaluated as logical AND conditions, therefore the target devices must satisfy all of them to use the media stream the attributes belong to.

D.7.1 CE manufacturer filter

This filter shall be implemented by all DVB compliant FuSs and CPEs.

Syntax

```
a=x-dvb-rms-ce-manufacturer:<ManufacturerOUI> … <ManufacturerOUI>
```

This attribute is the only one which can be used both in the session-description and in the media-description, therefore it will apply to all attribute lines. The `<ManufacturerOUI>` data items shall be separated by a space character.

Semantics

For each CE manufacturer who is represented in the announcement message their OUI shall be included in the attribute values.

EXAMPLE:

Message including SDP coding:

```
v=0
o=dvb-fus 2890844526 2890842807 IN IP4 10.47.16.5
s=dvb-update
t=2873397496 2873404696
a=x-dvb-rms-ce-manufacturer:00D09D 00D09E 00D09F
…
```

In this example, the announcement is for the set of CPEs which belongs to manufacturer whose OUI is 00D09D, 00D09E and 00D09F.

D.7.2 Device class filter

It is recommended that this filter is implemented by both FUS and CPE.

Syntax

```
a=x-dvb-rms-device-class: <ManufacturerOUI> <ProductClass> <HardwareVersion> <SoftwareVersion>
```

Semantics

As defined by the Metadata <TargetDevices><DeviceClass>…, the announcement is for CPE which matches the attribute values.

EXAMPLE:

Message including SDP and media coding:

```
v=0
o=dvb-fus 2890844526 2890842807 IN IP4 10.47.16.5
s=dvb-update
t=2873397496 2873404696
a=x-dvb-rms-ce-manufacturer:00D09D 00D09F
m=application 49153 FLUTE/UDP *
c=IN IP4 240.0.0.3
a=flute-tsi:142
a=source-filter:IN IP4 incl 192.168.1.1
a=x-dvb-rms-device-class:00D09D STB hw1.0 sw1.0
…
```

In this example the media-description is used to announce a FLUTE session delivery to the destination port 49153 and multicast IPv4 address 240.0.0.3. The FLUTE data stream identified by the TSI=142 will be sent from the address 192.168.1.1 and the target CPEs have OUI=00D09D, product class=STB, hardware version hw1.0 and software version sw1.0.

D.7.3 Device group filter

Several options are specified, based on structures defined in the Metadata schema <TargetDevices><DeviceGroup>…. They may be optionally supported by the FUS and CPE.

Syntax

```
a=x-dvb-rms-device-group:<Mode> <ManufacturerOUI> <ProductClass>

a=x-dvb-rms-device-group-hw: <Mode> <ManufacturerOUI> <ProductClass>
<HardwareVersion> … <HardwareVersion>

a=x-dvb-rms-device-group-sw: <Mode> <ManufacturerOUI> <ProductClass>
<SoftwareVersion> … <SoftwareVersion>

a=x-dvb-rms-device-group-sn-range: <Mode> <ManufacturerOUI> <ProductClass> <Mode>
<SerialNumberLowerBound> <SerialNumberUpperBound>

a=x-dvb-rms-device-group-sn-list: <Mode> <ManufacturerOUI> <ProductClass> <Mode> <SerialNumber> …
<SerialNumber>

a=x-dvb-rms-device-group-mac-range: <Mode> <ManufacturerOUI> <ProductClass> <Mode>
<MACAddressLowerBound> <MACAddressUpperBound>

a=x-dvb-rms-device-group-mac-list: <Mode> <ManufacturerOUI> <ProductClass> <Mode>
<MACAddress> … <MACAddress>
```

Semantics

The fields shall be included exactly as used in the metadata schema, and shall all be included in the order shown in the syntax above.

The combination of values used in the attribute fields shall be capable of exactly describing a population of CPEs.

The <Mode> field shall be included in all device group filter attributes and can be either "INCL" or "EXCL" to specify if the target CPEs shall match the following parameters or shall not match them.

Annex E (normative): DVB specific data model extensions required for DVB Content Download Service

NOTE: This annex contains normative information if the DVB data model extensions described in clause 7 are used.

This annex describes the management methods and additional data model objects for the storage management for DVB Content Download Services (DVB CDS) as defined in clause 10 of TS 102 034 [1].

DVB CDS supports push and pull download service modes. In the push download service mode the download of content items to the local storage of a HNED is initiated by the service provider, without explicit request by the user. In the pull download service mode the download is initiated at the explicit request from the user. Unicast and multicast download protocols are defined and can be used for both service modes. For details see TS 102 034 [1], clause 10.6.

Storage management is concerned with the management of the DVB CDS specific local storage on the HNED and the management of the content items that have been downloaded to this storage by CDS. For a definition of a content item see TS 102 034 [1], clause 10.4.

E.1 CDS Push Service storage management

CDS Push service storage is always managed by the service provider via RMS.

The following management methods shall be supported per individual HNED:

- Query if the HNED provides a CDS Push service storage space.
- Query and modify the overall CDS Push Service storage space.
- Query the used CDS Push Service storage space.
- Query the list of content items stored on the CDS Push Service storage space (this provides the content identifier and version number).
- Delete individual content items on the CDS Push Service storage space (this deletes all files that belong to the content item).

E.2 CDS Pull Service storage management

The CDS Pull Service storage is usually managed by the user. However by configuration from the user the service provider can be allowed to manage the storage via RMS.

The following management methods shall be supported per individual HNED:

- Query if the HNED provides a CDS Pull service storage space.
- Query if the CDS Pull Service storage management via RMS is enabled.

The following management methods apply only if CDS Pull Service storage management via RMS is enabled:

- Query and modify the overall CDS Pull Service storage space.
- Query the used CDS Pull Service storage space.
- Query the list of content items stored on the CDS Pull Service storage space (this provides the content identifier and version number).
- Delete individual content items on the CDS Pull Service storage space (this deletes all files that belong to the content item).

Annex F (normative): Extension of TR-069 / CWMP to carry DVB specific Content Download Service data model extensions

NOTE: This annex contains normative information if the DVB data model extensions described in clause 7 are used.

In this annex, the CWMP extensions to support the DVB Content Download System (CDS) service are considered. The DVB CDS is specified in clause 10 of [1].

No extensions to TR-069 are necessary. The extensions for the STB data model are based on TR-135 Issue 1 [8]. The TR-140 data model for storage service [5] is implicitly used since the new TR-135 objects defined in this annex refer to instances of TR-140 storage service objects.

The TR-069 data model for the IPTV STB, including all the CPE parameters needed for provisioning and assurance, may be specified as a superset of the available Broadband Forum data models, such as the generic device data model [7] and the IPTV specific data model [8].

F.1 Rationale for the chosen solution and guidelines for implementation

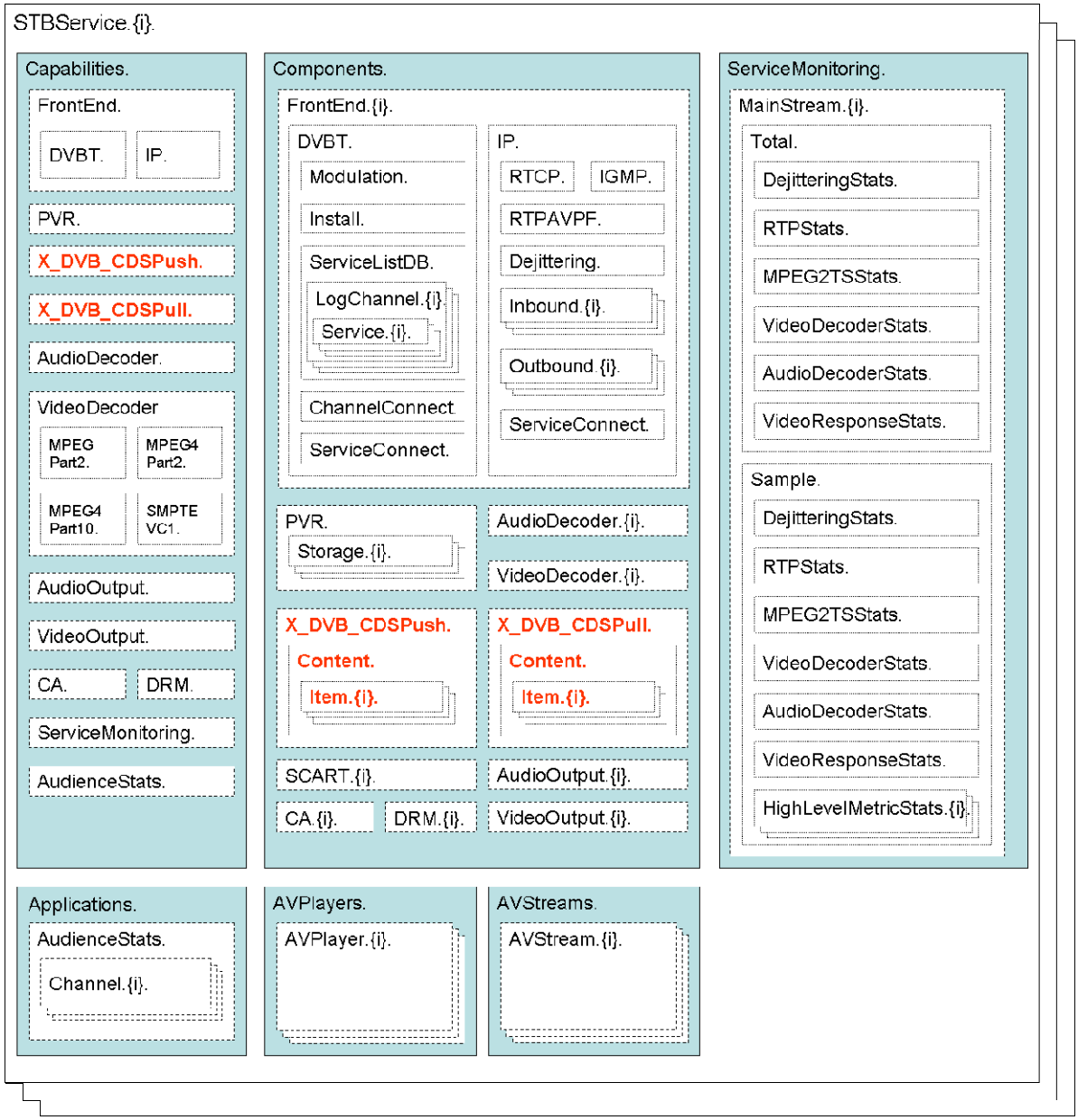

Figure F.1: TR-135 STBService Object Structure with DVB extensions

The DVB extensions for the STB data model are shown using red characters in figure F.1 which is based on figure 2 of TR-135 - STBService Object Structure [8].

According to TR-135 the PVR.Storage object is a multiple instance object. The single instances refer to either TR-140 [5] StorageService instance, or an object contained within such an instance, i.e. a PhysicalMedium, a LogicalVolume, or a Folder.

1) TR-135 PVR.Storage object refers to a StorageService instance:

   This is not really meaningful since the TR.140 StorageService also defines objects not referring to storage space but are related to applications such as a FTP server, etc. Therefore it is not recommended to implement the TR-135 PVR.Storage object referring to a StorageService instance.

2) TR-135 PVR.Storage object refers to a PhysicalMedium:

   The PhysicalMedium may contain multiple logical volumes and thus this reference might not be unique regarding where the PVR content is eventually stored. There are further disadvantages as the PhysicalMedium object only allows to provide the Capacity (the formatted capacity of the physical medium) and not information as used space, etc. Therefore it is not recommended to implement the TR-135 PVR.Storage object referring to a PhysicalMedium.

3) TR-135 PVR.Storage object refers to a LogicalVolume:

   Using this option allows to retrieve parameters such as the Capacity and UsedSpace of the PVR.Storage. As long as only the PVR service is supported on the STB this option is acceptable. However, there are some disadvantages if later on the CDS service will be added, e.g. at a later time supported through a firmware upgrade. If the CDS service would use the same StorageService as the PVR service, then storage demands of both services could conflict, e.g. no space left for CDS because the space is already occupied with PVR recordings. If the CDS service would use a different StorageService as the PVR service, then (since most likely unallocated storage space is not available) the PVR StorageService has to be deleted and to be newly created using less space. This would result in losing all PVR recordings.

4) TR-135 PVR.Storage object refers to a Folder:

   Using this option allows to retrieve parameters such as the Capacity and UsedSpace of the PVRStorage. If supported on the device Quota can be administered for the folder thus allowing to limit the storage space used for PVR. With this option the CDS service could use the same LogicalVolume and configuring Folder.Quota would ensure that storage demands of the services do not conflict and can be easily adapted via CWMP. Therefore it is recommended to implement the TR-135 PVR.Storage object referring to a Folder. Additionally support of Quota management is highly recommended.

DVB defines two new TR-135 objects to support the CDS service as illustrated in figure F.1 namely the CDSPush and CDSPull objects. In contrast to the PVR.Storage objects the CDS objects are single instance objects. For a detailed definition of those objects refer to clause A.5.2. It is recommended that the objects refer to separate Folder objects within a LogicaVolume.

An example explaining the partitioning of a STB hard disk is shown in the following figure. Please note that the names used for the devices, folders, etc. are for illustrative purposes only and thus are not mandatory.

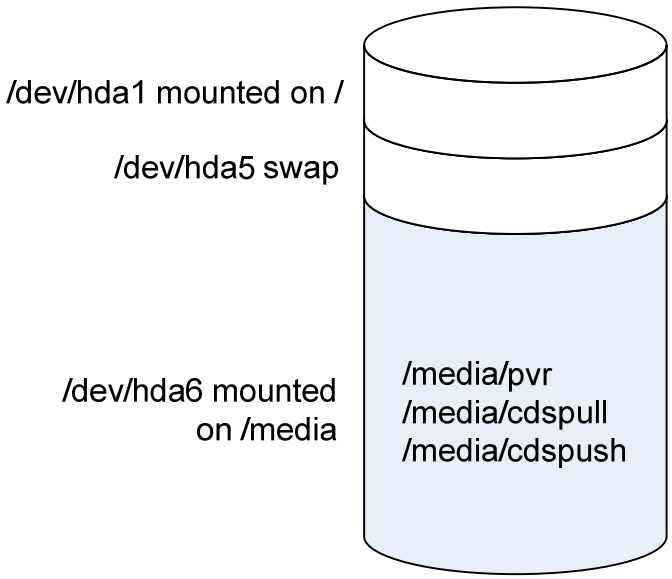

Figure F.2: Example of STB hard disk partitions

F.2 DVB data model extensions

This clause contains the DVB customization for the data model in order to support the content download service functionality. These extensions are not required to be standardized in Broadband Forum.

Table F.1: Parameter list for a STB CPE device for CDS support

Name	Type	Write	Description	Default
.STBService.{i}.Capability.-X_DVB_CDSPush.				
CDSPushCapable	boolean	-	Does this device provide a CDS Push service for the operator initiated download of content items to a local storage via a broadband IP connection?	
.STBService.{i}.Capability.-X_DVB_CDSPull.				
CDSPullCapable	boolean	-	Does this device provide a CDS Pull service for the user initiated download of content items to a local storage via a broadband IP connection?	
.STBService.{i}.Components.-X_DVB_CDSPush.	Object	-		
Reference	string(256)	-	References an object contained within a StorageService instance, e.g. a PhysicalMedium, LogicalVolume or Folder instance. The value is the full hierarchical name of the corresponding object. Example: "Device.-Services.StorageService.1.LogicalVolume.1.Folder.1".	
.STBService.{i}.Components.X_DVB_CDSPush.Content.	object	-		
ItemNumberOfEntries	unsignedInt	-	Number of Content Item instances.	
.STBService.{i}.Components.-X_DVB_CDSPush.-Content.Item.{i}.	object	W		
ContentReferenceId	string	-	Reference as e.g. defined in [39].	
VersionNumber	unsignedInt[0:255]	-		
.STBService.{i}.Components.-X_DVB_CDSPull.	Object	-		
OperatorManagedEnable	boolean	-	Enables or disables remote management of the CDSPull objects. If set to FALSE the device will not reveal any parameters belonging to the .STBService.{i}.Components.CDSPull object. If set to TRUE the .STBService.{i}.Components.CDSPull parameters are accessible via CWMP.	
Reference	string(256)	-		
.STBService.{i}.Components.-X_DVB_CDSPull.Content.	object	-		
ItemNumberOfEntries	unsignedInt	-	Number of Content Item instances.	
.STBService.{i}.Components.-X_DVB_CDSPull.-Content.Item.{i}.	object	W		
ContentReferenceId	string	-	Reference as e.g. defined in [39].	
VersionNumber	unsignedInt[0:255]	-		

F.3 Remote Management functionality using CWMP

In this clause it will be briefly explained how TR-069 remote management can be used to perform the basic tasks for the management of the DVB CDS service:

1) *RMS may read the CDS Push Service storage capacity/used space*: this can be done using the GetParameterValues RPC for the Components.X_DVB_CDSPush.Reference parameter to retrieve the full hierarchical name of the actual TR-140 StorageService instance or an object within this instance. A subsequent GetParameterValues RPC for the capacity/used space parameter of this StorageService object provides the requested information.

2) *RMS may modify the CDS Push Service storage capacity*: this can be done using the GetParameterValues RPC for the Components.X_DVB_CDSPush.Reference parameter to retrieve the full hierarchical name of the actual StorageService instance or an object within this instance. If the X_DVB_CDSPush.Reference parameter points to a TR-140 folder object, then a subsequent GetParameterNames RPC for the folder object would reveal whether quota are supported on the STB. If supported, then a SetParameterValue RPC for the Quota.Capacity parameter of the TR-140 StorageService referenced by an instance of the X_DVB_CDSPush.Reference defines the storage space.

3) *RMS may query a list of CDS Push Service content items*: this can be done using the GetParameterValues RPC using the partial path name Device.Service.STBService.1.Components.X_DVB_CDSPush.Content. Please note that this is an example only and concrete instance number may differ.

4) *RMS may delete individual CDS Push Service content items*: first get a list of content items as described in the previous item and then use a DeleteObject RPC for the appropriate content item object instance. As a consequence of the RPC the STB has to delete all files associated with the content item, e.g. movie file, meta data file, bookmarks file, etc.

In contrast to the CDS Push Service for CDS Pull the user has to explicitly agree that the operator is allowed to manage the service storage either as part of a contract or for help desk issues (user complains about missing storage space for further downloads). It is assumed that enabling of this capability is controlled via a local GUI displayed on the TV screen.

1) *RMS may read the CDS Pull Service storage space*: the procedure is identical to the procedure in item 1) except that the Components.X_DVB_CDSPull object should be used instead of Components. X_DVB_CDSPush. The user has to explicitly enable the operator for the CDS Pull Service management tasks. Whether the operator management is enabled can be queried by sending a GetParameterValues RPC for the Components.X_DVB_CDSPull.OperatorManagedEnable parameter.

2) *RMS may modify the CDS Pull Service storage space*: the procedure is identical to the procedure in item 2) except that the Components.X_DVB_CDSPull object should be used instead of Components. X_DVB_CDSPush. The user has to explicitly enable the operator for the CDS Pull Service management tasks. Whether the operator management is enabled can be queried by sending a GetParameterValues RPC for the Components.X_DVB_CDSPull.OperatorManagedEnable parameter.

3) *RMS may query a list of CDS Pull Service content items*: the procedure is identical to the procedure in item 3) except that the Components.X_DVB_CDSPull object should be used instead of Components. X_DVB_CDSPush. The user has to explicitly enable the operator for the CDS Pull Service management tasks. Whether the operator management is enabled can be queried by sending a GetParameterValues RPC for the Components.X_DVB_CDSPull.OperatorManagedEnable parameter.

4) *RMS may delete individual CDS Pull Service content items*: the procedure is identical to the procedure in item 4) except that the Components.X_DVB_CDSPull object should be used instead of Components. X_DVB_CDSPush. The user has to explicitly enable the operator for the CDS Pull Service management tasks. Whether the operator management is enabled can be queried by sending a GetParameterValues RPC for the Components.X_DVB_CDSPull.OperatorManagedEnable parameter.

History

Document history		
V1.1.1	July 2008	Publication
V1.2.1	February 2010	Publication