Data-Over-Cable Service Interface Specifications
DCA - MHAv2

Remote PHY OSS Interface Specification

CM-SP-R-OSSI-I03-160512

ISSUED

DISCLAIMER

This document is furnished on an "AS IS" basis and neither CableLabs nor its members provides any representation or warranty, express or implied, regarding the accuracy, completeness, noninfringement, or fitness for a particular purpose of this document, or any document referenced herein. Any use or reliance on the information or opinion in this document is at the risk of the user, and CableLabs and its members shall not be liable for any damage or injury incurred by any person arising out of the completeness, accuracy, or utility of any information or opinion contained in the document.

CableLabs reserves the right to revise this document for any reason including, but not limited to, changes in laws, regulations, or standards promulgated by various entities, technology advances, or changes in equipment design, manufacturing techniques, or operating procedures described, or referred to, herein.

This document is not to be construed to suggest that any company modify or change any of its products or procedures, nor does this document represent a commitment by CableLabs or any of its members to purchase any product whether or not it meets the characteristics described in the document. Unless granted in a separate written agreement from CableLabs, nothing contained herein shall be construed to confer any license or right to any intellectual property. This document is not to be construed as an endorsement of any product or company or as the adoption or promulgation of any guidelines, standards, or recommendations.

Document Status Sheet

Document Control Number:	CM-SP-R-OSSI-I03-160512			
Document Title:	Remote PHY OSS Interface Specification			
Revision History:	I01 - Released 08/17/2015 I02 - Released 01/21/2016 I03 - Released 05/12/2016			
Date:	May 12, 2016			
Status:	~~Work in Progress~~	~~Draft~~	**Issued**	~~Closed~~
Distribution Restrictions:	~~Author Only~~	~~CL/Member~~	~~CL/ Member/ Vendor~~	**Public**

Key to Document Status Codes

Work in Progress	An incomplete document, designed to guide discussion and generate feedback that may include several alternative requirements for consideration.
Draft	A document in specification format considered largely complete, but lacking review by Members and vendors. Drafts are susceptible to substantial change during the review process.
Issued	A generally public document that has undergone Member and Technology Supplier review, cross-vendor interoperability, and is for Certification testing if applicable. Issued Specifications are subject to the Engineering Change Process.
Closed	A static document, reviewed, tested, validated, and closed to further engineering change requests to the specification through CableLabs.

Trademarks

Contents

Figures

Tables

This page left blank intentionally.

1 SCOPE

1.1 Introduction and Purpose

This document describes the Operations Support System Interface (OSSI) for the Modular Headend Architecture version 2 (MHAv2). MHAv2 is initially targeted to permit a CMTS to support an IP-based digital HFC plant. In an IP-based digital HFC plant, the fiber portion utilizes a baseband network transmission technology such as Ethernet, EPON (Ethernet Passive Optical Network), GPON (Gigabit Passive Optical Network), or any Layer 2 technology that would support a fiber-based layer 1.

MHAv2 uses a layer 3 pseudowire between a CCAP Core and a series of Remote PHY devices. One of the common locations for a Remote PHY device is at an optical node at the junction of the fiber and coax plants.

1.2 MHAv2 Interface Documents

A list of the documents in the MHAv2 family of specifications is provided below. For updates, refer to http://www.cablelabs.com/specs/specification-search/?cat=docsis&scat=dca-mhav2.

Designation	Title
[R-PHY]	Remote PHY Specification
[R-DEPI]	Remote Downstream External PHY Interface Specification
[R-UEPI]	Remote Upstream External PHY Interface Specification
[GCP]	Generic Control Plane Specification
[R-DTI]	Remote DOCSIS Timing Interface Specification
[R-OOB]	Remote Out-of-Band Specification
R-OSSI (this document)	Remote PHY Operations Support System Interface Specification

MHAv2 does not explicitly use the DTI specification or any of the MHA specifications.

1.3 Requirements

Throughout this document, the words that are used to define the significance of particular requirements are capitalized. These words are:

"MUST"	This word means that the item is an absolute requirement of this specification.
"MUST NOT"	This phrase means that the item is an absolute prohibition of this specification.
"SHOULD"	This word means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before choosing a different course.
"SHOULD NOT"	This phrase means that there may exist valid reasons in particular circumstances when the listed behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
"MAY"	This word means that this item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.

This document defines many features and parameters, and a valid range for each parameter is usually specified. Equipment (CMTS and CCAP) requirements are always explicitly stated. Equipment complying with all mandatory

(MUST and MUST NOT) requirements is considered compliant with this specification. Support of non-mandatory features and parameter values is optional.

1.4 Conventions

In this specification the following convention applies any time a bit field is displayed in a figure. The bit field should be interpreted by reading the figure from left to right, then from top to bottom, with the MSB being the first bit so read and the LSB being the last bit so read.

```
MIB syntax, XML Schema and YANG module syntax are represented by this code sample font.
```

NOTE: Notices and/or Warnings are identified by this style font and label.

2 REFERENCES

2.1 Normative References[1]

In order to claim compliance with this specification, it is necessary to conform to the following standards and other works as indicated, in addition to the other requirements of this specification. Notwithstanding, intellectual property rights may be required to use or implement such normative references.

[CANN]	CableLabs Assigned Names and Numbers, CL-SP-CANN-I14-160317, March 17, 2016, Cable Television Laboratories, Inc.
[CCAP-CONFIG-YANG]	CCAP YANG Configuration Module, rphy@2016-04-21.yang, http://www.cablelabs.com/YANG/DOCSIS/rphy/
[CCAP-EVENTS-YANG]	CCAP YANG Module for Event Messaging, CCAPevents.yang, http://www.cablelabs.com/YANG/DOCSIS
[CCAP-OSSIv3.1]	DOCSIS 3.1 CCAP OSSI Specification, CM-SP-CCAP-OSSIv3.1-I06-151210, December 10, 2015, Cable Television Laboratories, Inc.
[CLAB-DEF-MIB]	CableLabs Definition MIB Specification, CL-SP-MIB-CLABDEF-I12-160325, March 25, 2016, Cable Television Laboratories, Inc.
[CLAB-TOPO-MIB]	CableLabs Topology MIB, CLAB-TOPO-MIB, http://www.cablelabs.com/MIBs/common/.
[DOCS-BPI2EXT-MIB]	DOCSIS Baseline Privacy Plus Extension MIB Module, DOCS-BPI2EXT-MIB, http://www.cablelabs.com/MIBs/DOCSIS/
[DOCS-DIAG-MIB]	DOCSIS Diagnostic Log MIB, DOCS-DIAG-MIB, http://www.cablelabs.com/MIBs/DOCSIS/.
[DOCS-IF3-MIB]	DOCSIS Interface 3 MIB Module, DOCS-IF3-MIB, http://www.cablelabs.com/MIBs/DOCSIS/.
[DOCS-IFEXT2-MIB]	DOCSIS Interface Extension 2 MIB Module, DOCS-IFEXT2-MIB, http://www.cablelabs.com/MIBs/DOCSIS/.
[DOCS-PNM-MIB]	DOCSIS PNM MIB Module, DOCS-PNM-MIB, http://www.cablelabs.com/MIBs/DOCSIS/.
[DOCS-RPHY-MIB]	DOCSIS Remote PHY MIB Module, DOCS-RPHY-MIB, http://www.cablelabs.com/MIBs/DOCSIS/.
[3DES]	Federal Information Processing Standards Publications 46-3, Data Encryption Standard (DES), October 25, 1999, http://csrc.nist.gov/publications/fips/archive/fips46-3/fips46-3.pdf.
[FIPS-180]	Federal Information Processing Standards Publications 180-4, Secure Hash Standard, August 2015.
[FIPS-197]	Federal Information Processing Standards Publications 197, Specification for the Advanced Encryption Standard (AES), November 26, 2001.
[GCP]	Generic Control Plane Specification, CM-SP-GCP-I02-160512-May 12, 2016, Cable Television Laboratories, Inc
[ISO 6709]	ISO 6709:2008, Standard representation of geographic point location by coordinates.
[L2VPN]	Layer 2 Virtual Private Networks, CM-SP-L2VPN-I15-150528, May 28, 2015, Cable Television Laboratories, Inc.
[MULPIv3.1]	MAC and Upper Layer Protocols Interface Specification, CM-SP-MULPIv3.1-I08-151210, December 10, 2015, Cable Television Laboratories, Inc.
[OSSIv3.0]	Operations Support System Interface Specification, CM-SP-OSSIv3.0-I28-151210, December 10, 2015, Cable Television Laboratories, Inc.

[1] Modified per R-OSSI-N-15.1411-3 on 1/8/15 by KB.

[PHYv3.1] DOCSIS Physical Layer Specification, CM-SP-PHYv3.1-I08-151210, December 10, 2015, Cable Television Laboratories, Inc.

[R-DEPI] Remote Downstream External PHY Interface Specification, CM-SP-R-DEPI-I04-160512-May 12, 2016, Cable Television Laboratories, Inc.

[R-DTI] Remote DOCSIS Timing Interface Specification, CM-SP-R-DTI-I02-151001, October 1, 2015, Cable Television Laboratories, Inc.

[RFC 1350] IETF RFC 1350/STD0033, The TFTP Protocol (Revision 2), July 1992.

[RFC 2560] IETF RFC 2560, X.509 Internet Public Key Infrastructure Online Certification Status Protocol - OCSP, June 1999.

[RFC 2573] IETF RFC 2573, SNMP Applications, April 1999.

[RFC 2575] IETF RFC 2575, View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP), April 1999.

[RFC 2578] IETF RFC 2578, Structure of Management Information Version 2 (SMIv2), April 1999.

[RFC 2669] IETF RFC 2669, DOCSIS Cable Device MIB Cable Device Management Information Base for DOCSIS compliant Cable Modems and Cable Modem Termination Systems, August 1999.

[RFC 2786] IETF RFC 2786, Diffie-Helman USM Key Management, March 2000.

[RFC 2790] IETF RFC 2790, Host Resources MIB, March 2000.

[RFC 2856] IETF RFC 2856, Textual Conventions for Additional High Capacity Data Types, June 2000.

[RFC 2863] IETF RFC 2863, The Interfaces Group MIB, June 2000.

[RFC 3164] IETF RFC 3164, The BSD Syslog Protocol, August 2001.

[RFC 3289] IETF RFC 3289, Management Information Base for the Differentiated Services Architecture, June 2002.

[RFC 3412] IETF RFC 3412, Message Processing and Dispatching for the Simple Network Management Protocol (SNMP), December 2002.

[RFC 3418] IETF RFC 3418/STD0062, Management Information Base (MIB) for the Simple Network Management Protocol (SNMP), December 2002.

[RFC 3433] IETF RFC 3433, Entity Sensor Management Information Base, December 2002.

[RFC 3584] IETF RFC 3584, Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework, August 2003.

[RFC 3635] IETF RFC 3635, Definitions of Managed Objects for the Ethernet-like Interface Types, October 2003.

[RFC 3931] IETF RFC 3931, Layer Two Tunneling Protocol - Version 3 (L2TPv3), March 2005

[RFC 3986] IETF RFC 3986, Uniform Resource Identifier (URI): Generic Syntax, January 2005.

[RFC 4022] IETF RFC 4022, Management Information Base for the Transmission Control Protocol (TCP), March 2005.

[RFC 4113] IETF RFC 4113, Management Information Base for the User Datagram Protocol (UDP), June 2005.

[RFC 4131] IETF RFC 4131, Management Information Base for Data Over Cable Service Interface Specification (DOCSIS) Cable Modems and Cable Modem Termination Systems for Baseline Privacy Plus, September 2005.

[RFC 4133] IETF RFC 4133, Entity MIB (Version 3), August 2005.

[RFC 4188] IETF RFC 4188, Definitions of Managed Objects for Bridges, September 2005.

[RFC 4250] IETF RFC 4250, The Secure Shell (SSH) Protocol Assigned Numbers, January 2006.

[RFC 4251] IETF RFC 4251, The Secure Shell (SSH) Protocol Architecture, January 2006.

[RFC 4252] IETF RFC 4252, The Secure Shell (SSH) Authentication Protocol, January 2006.

[RFC 4253] IETF RFC 4253, The Secure Shell (SSH) Transport Layer Protocol, January 2006.

[RFC 4254] IETF RFC 4254, The Secure Shell (SSH) Connection Protocol, January 2006.

[RFC 4293] IETF RFC 4293, Management Information Base for the Internet Protocol (IP), April 2006.

[RFC 4546] IETF RFC 4546, Radio Frequency (RF) Interface Management Information Base for Data over Cable Service Interface Specifications (DOCSIS) 2.0 Compliant RF Interfaces, June 2006.

[RFC 4639] IETF RFC 4639, Cable Device Management Information Base for Data-Over-Cable Service Interface Specification (DOCSIS) Compliant Cable Modems and Cable Modem Termination Systems, December 2006.

[RFC 5277] IETF RFC 5277, NETCONF Event Notifications, July 2008.

[RFC 5280] IETF RFC 5280, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, May 2008.

[RFC 5601] IETF RFC 5601, Pseudowire (PW) Management Information Base (MIB), July 2009.

[RFC 5612] IETF RFC 5612, Enterprise Number for Documentation Use, August 2009.

[RFC 6021] IETF RFC 6021, Common YANG Data Types, October 2010.

[RFC 6668] IETF RFC 6668, SHA-2 Data Integrity Verification for the Secure Shell (SSH) Transport Layer Protocol.

[RFC 6991] IETF RFC 6991, Common YANG Data Types, July 2013.

[R-OOB] Remote Out-of-Band Specification, CM-SP-R-OOB-I03-160512, May 12, 2016, Cable Television Laboratories, Inc.

[R-PHY] Remote PHY System Specification, CM-SP-R-PHY-I04-160512, May 12, 2016, Cable Television Laboratories, Inc.

[R-UEPI] Remote Upstream External PHY Interface Specification, CM-SP-R-UEPI-I03-160512, May 12, 2016, Cable Television Laboratories, Inc.

[SCTE 154-2] ANSI SCTE 154-2 2008, SCTE-HMS-QAM-MIB.

[SCTE 154-5] ANSI SCTE 154-5 2008, SCTE-HMS-HEADENDIDENT TEXTUAL CONVENTIONS MIB.

[SECv3.1] DOCSIS 3.1 Security Specification, CM-SP-SECv3.1-I05-151210, December 10, 2015, Cable Television Laboratories, Inc.

[W3 XML1.0] Extensible Markup Language (XML) 1.0 (Third Edition), W3C Recommendation 04, February 2004.

[W3 XSD1.0] XML Schema Part 1: Structures Second Edition, W3C Recommendation 28, October 2004.

2.2 Informative References[2]

This specification uses the following informative references.

[CCAP TR] Converged Cable Access Platform Architecture Technical Report, CM-TR-CCAP-V03-120511, May 11, 2012, Cable Television Laboratories, Inc.

[DRFI] DOCSIS Downstream RF Interface Specification, CM-SP-DRFI- I14-131120, November 20, 2013, Cable Television Laboratories, Inc.

[IEEE 802.3x] 802.3x-1997 - IEEE Standards for Local and Metropolitan Area Networks: Specification for 802.3 Full Duplex Operation

[2] Modified per R-OSSI-N-15.1411-3 on 1/8/15 by KB.

[ISO 11404] BS ISO/IEC 11404:1996 Information technology--Programming languages, their environments and system software interfaces--Language-independent datatypes, January 2002.

[ISO 19501] ISO/IEC 19501:2005, Information technology - Open Distributed Processing - Unified Modeling Language (UML) Version 1.4.2.

[ITU-T X.692] ITU-T Recommendation X.692 (03/2002), Information technology - ASN.1 encoding rules: Specification of Encoding Control Notation (ECN).

[ITU-T M.3400] ITU-T Recommendation M.3400 (02/2000): TMN AND Network Maintenance: International Transmission Systems, Telephone Circuits, Telegraphy, Facsimile and Leased Circuits, TMN management functions.

[M-OSSI] DOCSIS M-CMTS Operations Support Interface, CM-SP-M-OSSI-I08-081209, December 9, 2008, Cable Television Laboratories, Inc.

[NSI] Cable Modem Termination System - Network Side Interface Specification, SP-CMTS-NSI-I01-960702, July 2, 1996, Cable Television Laboratories, Inc.

[PMI] Edge QAM Provisioning and Management Interface Specification, CM-SP-EQAM-PMI-I02-111117, November 17, 2011, Cable Television Laboratories, Inc.

[RFC 791] IETF RFC 791, Internet Protocol, September 1981.

[RFC 1042] IETF RFC 1042/STD0043, Standard for the transmission of IP datagrams over IEEE 802 networks, February 1988.

[RFC 1123] IETF RFC 1123/STD0003, Requirements for Internet Hosts - Application and Support, October 1989.

[RFC 1157] IETF RFC 1157, Simple Network Management Protocol (SNMP), May 1990.

[RFC 1213] IETF RFC 1213/STD17, Management Information Base for Network Management of TCP/IP-based internets: MIB-II, March 1991.

[RFC 1901] IETF RFC 1901, Introduction to Community-based SNMPv2, January 1996.

[RFC 2579] IETF RFC 2579, Textual Conventions for SMIv2, April 1999.

[RFC 2580] IETF RFC 2580, Conformance Statements for SMIv2, April 1999.

[RFC 3260] IETF RFC 3260, New Terminology and Clarifications for Diffserv, April 2002.

[RFC 3339] IETF RFC 3339, Date and Time on the Internet: Timestamps, July 2002.

[RFC 3410] IETF RFC 3410, Introduction and Applicability Statements for Internet-Standard Management Framework, December 2002.

[RFC 3411] IETF RFC 3411/STD0062, An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks, December 2002.

[RFC 3413] IETF RFC 3413, Simple Network Management Protocol (SNMP) Applications, December 2002.

[RFC 3414] IETF RFC 3414/STD0062, User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3), December 2002.

[RFC 3415] IETF RFC 3415, View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP), December 2002.

[RFC 3416] IETF RFC 3416, Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP), December 2002.

[RFC 3417] IETF RFC 3417, Transport Mappings for the Simple Network Management Protocol (SNMP), December 2002.

[RFC 3419] IETF RFC 3419, Textual Conventions for Transport Addresses, December 2002.

[RFC 4001] IETF RFC 4001, Textual Conventions for Internet Network Addresses, February 2005.

[RFC 4181] IETF RFC 4181, Guidelines for Authors and Reviewers of MIB Documents, September 2005.

[RFC 4291] IETF RFC 4291, IP Version 6 Addressing Architecture, February 2006.

[RFC 6020] IETF RFC 6020, YANG - A data modeling language for the Network Configuration Protocol (NETCONF), October 2010.

[SP 800-131] NIST Special Publication 800-131 A, Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths, Revision 1 draft, July 2015.

2.3 Reference Acquisition

- Cable Television Laboratories, Inc., 858 Coal Creek Circle, Louisville, CO 80027; Phone +1-303-661-9100; Fax +1-303-661-9199; http://www.cablelabs.com
- IANA, Internet Assigned Numbers Authority (IANA); http://www.iana.org
- IETF, Internet Engineering Task Force (IETF) Secretariat, 48377 Fremont Blvd., Suite 117, Fremont, California 94538, USA; Phone: +1-510-492-4080, Fax: +1-510-492-4001; http://www.ietf.org/
- ISO Specifications, International Organization for Standardization (ISO), 1, rue de Varembé, Case postale 56, CH-1211 Geneva 20, Switzerland; Phone +41 22 749 01 11; Fax +41 22 733 34 30; http://www.iso.org
- ITU Recommendations, International Telecommunication Union, Place des Nations, CH-1211, Geneva 20, Switzerland; Phone +41-22-730-51-11; Fax +41-22-733-7256; http://www.itu.int
- SCTE, Society of Cable Telecommunications Engineers Inc., 140 Philips Road, Exton, PA 19341; Phone: 1+610-363-6888 /1+ 800-542-5040; Fax: 1+610-363-5898; http://www.scte.org/
- World Wide Web Consortium (W3C), Massachusetts Institute of Technology, 32 Vassar Street, Room 32-G515, Cambridge, MA 02139; Phone +1-617-253-2613, Fax +1-617-258-5999; http://www.w3.org/Consortium/

3 TERMS AND DEFINITIONS

This specification uses the following terms:

Aggregation	A special type of object association for Configuration Object Models in which objects are assembled or configured together to create a more complex object.
Bonded Channels	A logical channel comprising multiple individual channels.
Bridging CMTS	A CMTS that makes traffic forwarding decisions between its Network Systems Interfaces and MAC Domain Interfaces based upon the Layer 2 Ethernet MAC address of a data frame.
Cable Modem	A modulator-demodulator at subscriber locations intended for use in conveying data communications on a cable television system.
Cable Modem Termination System	An access-side networking element or set of elements that includes one or more MAC Domains and one or more Network System Interfaces. This unit is located at the cable television system headend or distribution hub and provides data connectivity between a DOCSIS Radio Frequency Interface and a wide-area network.
Cable Modem Termination System - Network Side Interface (CMTS-NSI)	The interface, defined in [NSI], between a CMTS and the equipment on its network side.
Carrier-to-Noise plus Interference Ratio (CNIR)	The ratio of the expected commanded received signal power at the CMTS input to the noise plus interference in the channel.
CCAP Core	A CCAP device which uses MHAv2 protocols to interconnect to R-PHY Entity devices.
Channel	The frequency spectrum occupied by a signal. Usually specified by center frequency and bandwidth parameters.
Command Line Interface	A mechanism used to interact with the CCAP by typing text-based commands into a system interface.
Configuration Objects	Managed objects in the CCAP configuration that support writeability. The CCAP is configured by specifying the attributes of these objects.
Converged Cable Access Platform	An access-side networking element or set of elements that combines the functionality of a CMTS with that of an Edge QAM, providing high-density services to cable subscribers.
Converged Interconnect Network	The network (generally gigabit Ethernet) that connects a CCAP Core to an R-PHY Entity.
Customer Premises Equipment	Equipment at the end user's premises; may be provided by the service provider.
Downstream	1. Transmissions from CMTS to CM. This includes transmission from the CCAP Core to the RPD, as well as the RF transmissions from the RPD to the CM. 2. RF spectrum used to transmit signals from a cable operator's headend or hub site to subscriber locations.
Extensible Markup Language	A universal file format for storing and exchanging structured data. The CCAP configuration file is created in XML and has a specific schema, generated from a set of YANG modules, which are a physical implementation of an object model created to describe CCAP configuration.
FCAPS	A set of principles for managing networks and systems, wherein each letter represents one principle. F is for Fault, C is for Configuration, A is for Accounting, P is for Performance, and S is for Security.

Flow	A stream of packets in DEPI used to transport data of a certain priority from the CCAP Core to a particular QAM channel of the R-PHY Entity. In PSP operation, there can exist several flows per QAM channel.
Generalization	A relationship in which one configuration model element (the child) is based on another model element (the parent). A generalization relationship indicates that the child receives all of the attributes, operations, and relationships that are defined in the parent.
Hybrid Fiber/Coax System	A broadband bidirectional shared-media transmission system using optical fiber trunks between the headend and the fiber nodes, and coaxial cable distribution from the fiber nodes to the customer locations.
Institute of Electrical and Electronic Engineers	A voluntary organization which, among other things, sponsors standards committees and is accredited by the American National Standards Institute (ANSI).
Internet Engineering Task Force	A body responsible for, among other things, developing standards used in the Internet.
Internet Protocol	An Internet network-layer protocol.
L2TP Pseudowire (PW)	An emulated circuit as it traverses a packet-switched network. There is one Pseudowire per L2TP Session.
L2TP Pseudowire Type	The payload type being carried within an L2TP session. Examples include PPP, Ethernet, and Frame Relay.
L2TP Session	An L2TP session is the entity that is created between two LCCEs in order to exchange parameters for and maintain an emulated L2 connection. Multiple sessions may be associated with a single Control Connection.
MAC Domain	A grouping of Layer 2 devices that can communicate with each other without using bridging or routing. In DOCSIS, it is the group of CMs that are using upstream and downstream channels linked together through a MAC forwarding entity.
MAC Domain Cable Modem Service Group	The subset of a Cable Modem Service Group which is confined to the Downstream Channels and Upstream Channels of a single MAC domain. Differs from a CM-SG only if multiple MAC domains are assigned to the same CM-SGs.
Management	Functions on the CCAP that monitor for faults and for overall system performance, including traps and alarms.
Media Access Control	Used to refer to the Layer 2 element of the system which would include DOCSIS framing and signaling.
Management Information Base	A database of device configuration and performance information which is acted upon by SNMP.
Multiple System Operator	A corporate entity that owns and/or operates more than one cable system.
Open Systems Interconnection (OSI)	A framework of ISO standards for communication between different systems made by different vendors, in which the communications process is organized into seven different categories that are placed in a layered sequence based on their relationship to the user. Each layer uses the layer immediately below it and provides a service to the layer above. Layers 7 through 4 deal with end-to-end communication between the message source and destination, and layers 3 through 1 deal with network functions.
Physical (PHY) Layer	Layer 1 in the Open System Interconnection (OSI) architecture; the layer that provides services to transmit bits or groups of bits over a transmission link between open systems and which entails electrical, mechanical and handshaking procedures.

Quadrature Amplitude Modulation	A modulation technique in which an analog signal's amplitude and phase vary to convey information, such as digital data.
QAM Channel	Analog RF channel that uses quadrature amplitude modulation (QAM) to convey information.
Radio Frequency	In cable television systems, this refers to electromagnetic signals in the range 5 to 1000 MHz.
Remote PHY Device	The Remote PHY Device contains mainly PHY related circuitry, such as downstream QAM modulators, upstream QAM demodulators, and pseudowire logic to connect to the CCAP Core. Together, the CCAP Core and the R-PHY Entity are the functional equivalent of an I-CMTS (Integrated CMTS), just with different packaging.
Request for Comments	A technical policy document of the IETF; these documents can be accessed at http://www.rfc-editor.org/.
Routing CMTS	A CMTS that makes traffic forwarding decisions between its Network System Interfaces and MAC Domain Interfaces based upon the Layer 3 (network) address of a packet.
Running-config	Configuration objects that control CCAP behavior, along with any vendor-proprietary configurations.
Secure Copy Protocol	A secure file transfer protocol based on Secure Shell (SSH).
Simple Network Management Protocol	Allows a host to query modules for network-related statistics and error conditions.
Specialization	A relationship in which one configuration model element (the parent) is used to model another element (the child). The specialized child element receives all of the attributes, operations, and relationships that are defined in the parent and defines additional attributes, operations and relationships that enable its specialized behavior.
Startup-config	The configuration objects stored in non-volatile memory.
Upstream	1. Transmissions from CM to CCAP. This includes transmission from the RPD to the CCAP Core, as well as the RF transmissions from the CM to the RPD. 2. RF spectrum used to transmit signals from a subscriber location to a cable operator's headend or hub site.
X.509	ITU-T Recommendation standard for a public key infrastructure (PKI) for single sign-on (SSO) and Privilege Management Infrastructure (PMI).
YANG	A data modeling language for the NETCONF network configuration protocol. Though the CCAP physical data model for configuration makes use of one or more YANG modules, NETCONF implementation is not required for the integrated CCAP.

4 ABBREVIATIONS, ACRONYMS, AND NAMESPACES

This specification uses the following abbreviations:

AAA	Network Authentication, Authorization, and Accounting
ACL	Access Control List
AQM	Active Queue Management
AVP	Attribute Value Pair
BPI	Baseline Privacy Interface
BSS	Business Support Systems
CA	Certificate Authority
CCAP	Converged Cable Access Platform
CIN	Converged Interconnect Network
CLI	Command Line Interface
CM	Cable Modem
CMTS	Cable Modem Termination System
CPAF	Configuration, Performance, Accounting, Fault Management
CPE	Customer Premises Equipment
CRL	Certificate Revocation List
CW	Control Word
DBG	Downstream Bonding Group
DCID	Downstream Channel Identifier
DCS	Downstream Channel Sets
DEPI	Downstream External PHY Interface
DHCP	Dynamic Host Configuration Protocol
DLC	Downstream Line Card
DPoE	DOCSIS Provisioning of EPON
DS	Downstream
DSID	Downstream Service ID
EAE	Early Authentication and Encryption
EPON	Ethernet Passive Optical Network
EQAM	Edge QAM
ERM	Edge Resource Manager
ERMI	Edge Resource Manager Interface
ERRP	Edge Resource Registration Protocol
FCAPS	Fault, Configuration, Accounting, Performance and Security
FQDN	Fully Qualified Domain Name
FRU	Field Replaceable Unit
GCP	Generic Control Plane
HFC	Hybrid Fiber/Coax System
HTTPS	Secure Hypertext Transfer Protocol
I-CMTS	Integrated CMTS

IEEE	Institute of Electrical and Electronic Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol
ITU	International Telecommunication Union
L2VPN	Layer 2 Virtual Private Network
MAC	Media Access Control
MD-CM-SG	Media Access Control Domain Cable Modem Service Group
MDD	MAC Domain Descriptor
MIB	Management Information Base
MPEG	Moving Picture Experts Group
MPEG-TS	Moving Picture Experts Group-Transport Stream
MPT	MPEG-TS mode of DEPI
MPTS	Multi-Program Transport Stream
MSO	Multiple System Operator
MTC	Multiple Transmit Channel
MTU	Maximum Transmission Unit
NMS	Network Management System
NOC	Network Operations Center
NSI	Network Side Interface
OFDM	Orthogonal Frequency Division Multiplexing
OFDMA	Orthogonal Frequency Division Multiplexing with Multiple Access
OM	Object Model (Information Model)
OMG	Object Management Group
OOA&D	Object-Oriented Analysis and Design
OSS	Operations Support System
OSSI	Operations Support System Interface
OUI	Organization Unique Identifier
PAT	Program Association Table
PEN	Private Enterprise Number
PID	Packet Identifier
PLC	Phy Link Channel
PW	Pseudowire
QAM	Quadrature Amplitude Modulation
QoS	Quality of Service
QPSK	Quadrature Phase Shift Keying
RCC	Receive Channel Configuration
RCP	Receive Channel Profile
RDTI	Remote DTI
RF	Radio Frequency
RFC	Request for Comments
RPD	Remote PHY Device

R-PHY	Remote PHY
SA	Security Association
SCP	Secure Copy Protocol
SDV	Switched Digital Video
SMIv2	Structure of Management Information Version 2
SNMP	Simple Network Management Protocol
SNMPv1	Version 1 of the Simple Network Management Protocol
SNMPv2	Version 2 of the Simple Network Management Protocol
SNMPv3	Version 3 of the Simple Network Management Protocol
SSH	Secure Shell
SSM	Source Specific Multicast
STP	Spanning Tree Protocol
TCP	Transmission Control Protocol
TFTP	Trivial File Transfer Protocol
TLS	Transport Layer Security
TLV	Type Length Value Attribute
ToD	Time of Day
ToS	Terms of Service
TS	Transport Stream
TSID	Transport Stream Identifier
UBG	Upstream Bonding Group
UCID	Upstream Channel Identifier
UDC	Upstream Drop Classifier
UDP	User Datagram Protocol
ULC	Upstream Line Card
UML	Unified Modeling Language
URL	Uniform Resource Locator
US	Upstream
VLAN	Virtual Local Area Network
XML	Extensible Markup Language
XSD	XML Schema Definition

5 OVERVIEW

5.1 FCAPS Network Management Model

The International Telecommunication Union (ITU) Recommendation [ITU-T M.3400] defines a set of management categories, referred to as the FCAPS model, represented by the individual management categories of Fault, Configuration, Accounting, Performance and Security. Telecommunications operators, including MSOs, commonly use this model to manage large networks of devices. This specification uses these management categories to organize the requirements for the configuration and management of the Remote PHY platform.

Fault management seeks to identify, isolate, correct and record system faults. Configuration management modifies system configuration variables and collects configuration information. Accounting management collects usage statistics for subscribers, sets usage quotas and bills users according to their use of the system. Performance management focuses on the collection of performance metrics, analysis of these metrics and the setting of thresholds and rate limits. Security management encompasses identification and authorization of users and equipment, provides audit logs and alerting functions, as well as providing vulnerability assessment.

Each of these management categories is discussed in further detail in [CCAP-OSSIv3.1].

5.2 Management Architectural Overview

Figure 5-1 illustrates the Remote PHY management architecture. The CM, RPD and CCAP Core reside within the Network Layer where services are provided to end Subscribers and various metrics are collected about network and service performance, among other things. Various management servers reside in the Network Management Layer within the MSO back office to provision, monitor and administer the Network Elements within the Network Layer.

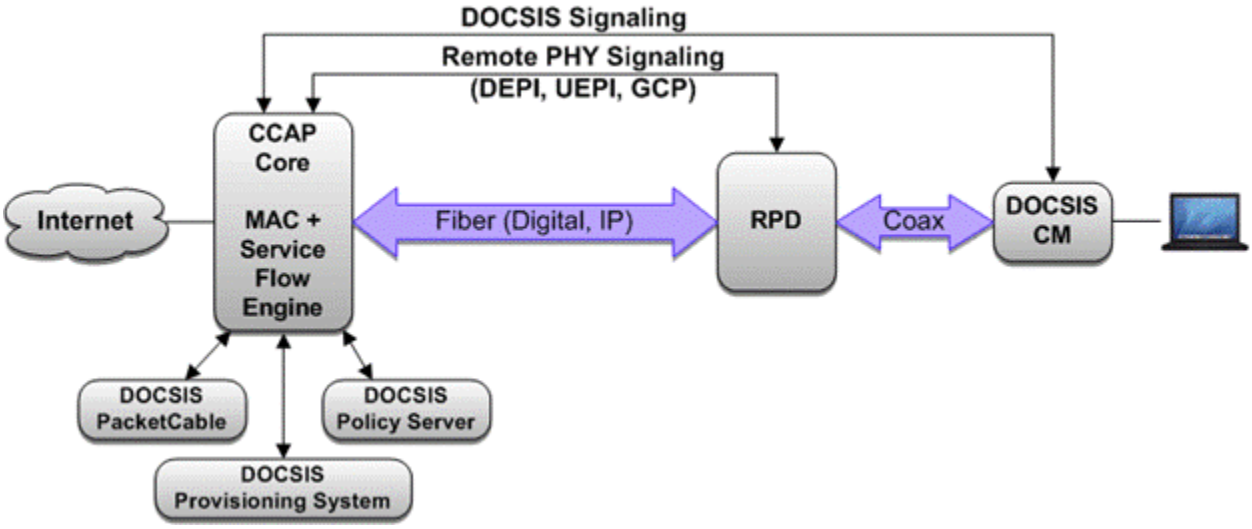

Figure 5-1 - CCAP Configuration Objects

Finally, the Business and Service Management Layer is where higher level MSO business processes are implemented via BSS/OSS systems. These BSS/OSS systems utilize the data and information from the Network Management Layer which interrogate data from the Network Layer.

5.3 Remote PHY OSSI Key Features

The primary goals of the Remote PHY OSSI are:

- Ensure that existing management interfaces on the CCAP are supported transparently to the NMS.

- Ensure that the RPD can be configured and managed both indirectly via the CCAP Core and, when necessary, directly.

Table 5-1 - Management Feature Requirements for Remote PHY

Features	Management Functional Area	OSI Layer	Description
RPD Configuration	Configuration	PHY, Data Link	Provisioning physical downstream and upstream interfaces and other features on the RPD indirectly via the CCAP Core.
CCAP Core Remote Phy Configuration	Configuration	PHY, Data Link	Configuration of the Remote PHY feature set and the RPD on the CCAP Core.
RPD Fault Detection	Fault	PHY, Network	Remote PHY Device fault definition and detection
CCAP Core Remote PHY Fault Detection	Fault	PHY, Network	Detection of faults related to the Remote PHY feature set on the CCAP Core.
RPD Performance/Status	Performance	PHY, Network	Interface for a management agent on the RPD to communicate non-DOCSIS performance and status information.
CCAP Core Performance/Status	Performance	PHY, Network	Monitoring of the DOCSIS Phy (indirectly via the RPD) and direct monitoring of non-DOCSIS performance and status information related to the Remote PHY feature set.

5.3.1 Fault Management Features

The Remote PHY Fault Management requirements include:

- Extended lists of events related to the new set of Remote PHY features for both the CCAP Core and RPD.
- Management requirements for the RPD to be managed directly, including an SNMP management agent (and associated MIBs) required on the RPD itself.
- Remote PHY Device (RPD) fault management requirements.
- Ensuring that existing faults defined for the CCAP function transparently in a Remote PHY environment.

5.3.2 Configuration Management Features

The configuration of the RPD by the CCAP Core is defined in this specification. The reporting of configuration state and status information is done via SNMP MIB objects. Configuration of features and functions of the CCAP is performed via XML configuration files.

The Remote PHY configuration requirements include:

- Configuration management of the CCAP Core as it relates to the Remote PHY Device and Remote PHY features.
- Configuration management of the Remote PHY by the CCAP Core.

5.3.3 Performance Management Features

The Remote PHY performance management requirements include:

- Non-DOCSIS performance management objects defined on the RPD and CCAP Core to monitor the performance and status of the Remote PHY feature.
- Ensuring that existing performance management objects defined for the CCAP function transparently in a Remote PHY environment.
- The CCAP Core controls and facilitates the Secure Software Download (SSD) for the RPD.[3]

[3] Added per R-OSSI-N-15.1409-2 on 2/3/15 by KB.

5.4 Information Models

The Information Model approach is based on an object-oriented modeling approach well known in the industry for capturing requirements and analyzing the data in a protocol independent representation. This approach defines requirements with use cases to describe the interactions between the operations support systems and the network element. The management information is represented in terms of objects along with their attributes and the interactions between these encapsulated objects (or also referred to as entities in some representations). The diagrams developed to capture these managed objects and their attributes and associations are UML Class Diagrams. The collection of UML Class Diagrams and Use Case Diagrams are referred to as the Remote PHY Information Models. With the introduction of several new, complex features in Remote PHY and the operator needs for a more proactive and efficient approach to management information, information modeling methodologies offer the ability to reuse the same definitions when new protocols are introduced in the future.

The managed objects are then represented in a protocol-specific form referred to as a Management Data Model. The Management Data Models when using SNMP are described using the Structure of Management Information Version 2 (SMIv2) [RFC 2578] and the design of these models is determined by the capabilities of the protocol. The Management Data Models when using XML configuration file download are described using XML Schema [W3 XSD1.0]. The Management Data Models when using GCP are defined as TLVs.

5.5 CCAP-OSSI Document Organization

This specification uses the FCAPS framework to group topics and content. In order to provide a more logical flow, one that mirrors processes in place at MSOs, the order of functions has been shifted and is organized as CPAF:

- Configuration Management
- Performance Management
- Accounting Management
- Fault Management

Note that Security Management topics are covered in context of these topics.

6 CONFIGURATION MANAGEMENT

In the Remote PHY architecture, the Remote PHY Device (RPD) has very minimal local configuration (e.g., certificates, passwords) and the majority of its operational configuration is provided during RPD initialization by the CCAP Core(s) via the control plane. Thus, operator configuration of the RPD is performed via configuration of the CCAP Core(s).

The R-PHY model supports configuration by a Primary CCAP Core and 0 or more Auxiliary CCAP Cores. This document will use the term CCAP Core to refer to either one.

6.1 RPD Configuration Theory of Operation

The CCAP controls its connected RPDs. A single CCAP serves as the single point of configuration for a set of resources (e.g., RF ports and channels) on a given RPD. The CCAP processes its configuration, which can include references to RPDs. Once the RPD bootstrap process has been completed, the CCAP translates applicable physical layer configuration to GCP objects. Then, the CCAP uses GCP to communicate these configuration objects to the relevant RPDs.

6.2 CCAP Configuration and Transport Protocol Requirements

6.2.1 Configuration Object Datastore

The CCAP supporting the Remote PHY architecture MUST implement the standard configuration objects defined by this specification.

The CCAP supporting the Remote PHY architecture MUST implement the standard configuration objects defined by [CCAP-OSSIv3.1], except where specified differently in this specification.

6.2.2 Dynamic Management of RPDs

When the downloaded XML-based configuration file contains information on a new RPD, the CCAP MUST provide the RPD configuration information via GCP to the new RPD.

When the downloaded XML-based configuration file contains information which modifies the configuration of a given RPD, the CCAP MUST utilize GCP to modify the RPD's configuration. The CCAP SHOULD do this in such a way as to minimize the impact of the change on other unchanged channels, ports, and functions on the RPD.

6.3 UML Configuration Object Model

6.3.1 CCAP UML Configuration Object Model Overview

For DOCSIS 3.0, 3.1, and DPoE, the CCAP UML configuration object model, as well as the schemas based on that object model, was divided into eight distinct groupings:

- CCAP: The Ccap object is the container of all CCAP configuration objects.
- Chassis: Consists of objects for configuring the hardware components of the CCAP.
- Video: Consists of those objects that are related to the EQAM functions of the CCAP, including ERM, encryption and decryption objects.
- DOCSIS: Consists of the DOCSIS configuration objects that are needed for configuring DOCSIS MAC Domains and services such as DSG.
- Network: Consists of objects related to configuring the core services for things like integrated servers, access lists, Syslog, HTTP, FTP, SSH, and other related network services.
- Interfaces: Consists of the objects needed to configure interfaces within the CCAP.
- Management: Consists of objects used to configure SNMP and Fault Management for the CCAP.
- EPON: Consists of the objects that are related to the DPoE configuration of the CCAP.

This specification extends that model to include an RPD grouping:

- RPD: Consists of objects that are related to the configuration of RPDs managed by the CCAP.

The CCAP supports the RPD-related configuration objects defined in the following sections via implementation of the CCAP XSD.

The CCAP configuration object model described in [CCAP-OSSIv3.1] has been modified in this specification for the Remote PHY architecture; those changes are described here. Objects not defined here are unchanged and detailed in [CCAP-OSSIv3.1].

6.3.1.1 Default Values and Mandatory Configuration of Attributes in the Configuration Object Model

In the configuration object model attribute tables in the following sections, a default value is defined in the Default table column for some object attributes. In cases where a default value is defined for an element, the CCAP will use the specified default value if the XML configuration file does not include the attribute.

In cases where the Default column reads "vendor-specific", the CCAP provides a default value of the vendor's choosing for the attribute in the implementation. In cases where the vendor is defining the default value, the operator need not include these attributes in the XML configuration file.

Attributes explicitly required in the XML configuration file are marked "Yes" in the Required Attribute column; these attributes do not have a default value. In these cases the operator needs to provide a value for these attributes in the XML configuration file when an object containing those attributes is being configured. In cases where the Required Attribute column reads "No", either a default value is provided in the table or the CCAP will provide a vendor-specific value.

6.3.1.2 Enumeration Values in the Configuration Object Model

In the configuration object model attribute tables in the following sections, enumerated lists are all intended to begin at a value of "1"; in most cases, the first value will be other ("other(1)"). Since this specification borrows objects from existing MIBs, there will be cases where the enumeration values specified here do not match those of the MIB on which the object attribute was based. CCAP vendors are expected to properly translate values provided in the XML configuration file into the correct values needed for SNMP reporting via the standard MIB objects.

Note that integers are specified for each enumeration in the UML configuration object model. When the UML is translated into other formats (XSD, YANG, SNMP, etc.), the enumeration labels and/or integers are included in these outputs as appropriate. For XSD and YANG, enumeration labels will be included.

6.3.1.3 Use of Interface Names in Configuration

Several configuration objects defined in this specification are identified with keys in the form of a text string name. In general, these configuration objects are modeled after interfaces that have equivalent representation in SNMP (ifTable). While this specification does not impose formal requirements on the format of interface names, CCAP vendors are expected to implement consistent conventions for assigning textual names to interfaces and disclose the rules on which such conventions are based. The CCAP typically rejects a configuration that includes an interface name that does not follow the vendor's naming conventions.

6.3.1.4 Unconstrained Strings in the Configuration Object Model

For object attributes with a data type of String, there are cases where this specification does not provide a length constraint. For these attributes, the CCAP can impose a vendor-specific length constraint. If a value in the XML configuration file exceeds this vendor-specific length constraint, the CCAP typically truncates the text string to that limit and logs an error.

6.3.2 Vendor-Specific Extensions

A CCAP is expected to implement vendor-proprietary configuration objects beyond those defined in this specification. Standard objects are those that have been defined in the configuration UML object model, defined in the following sections. Vendor-proprietary configuration objects consist of both new configuration objects not

represented in the CCAP configuration UML object model and new or modified attributes of configuration objects that exist in the CCAP configuration UML object model.

The CCAP's configuration object model can be extended via the creation of vendor-proprietary XSD schemas and/or vendor-proprietary YANG modules. A valid approach to vendor extensions is to perform extensions solely in XML schema utilizing the extension points in the standard schema (Additional details are expected in a future version of the specification.) in conjunction with a vendor-defined schema. Vendor extensions can also be performed in YANG. A CCAP that supports vendor extension in YANG also supports configuration via an XML configuration file based on an XSD schema that is the result of the conversion of the standard YANG module with extensions.

6.4 CCAP Configuration Objects

The CCAP configuration object model has been modified for the Remote PHY architecture; those changes are described in the following subsections. Objects not defined here are unchanged and are detailed in [CCAP-OSSIv3.1].

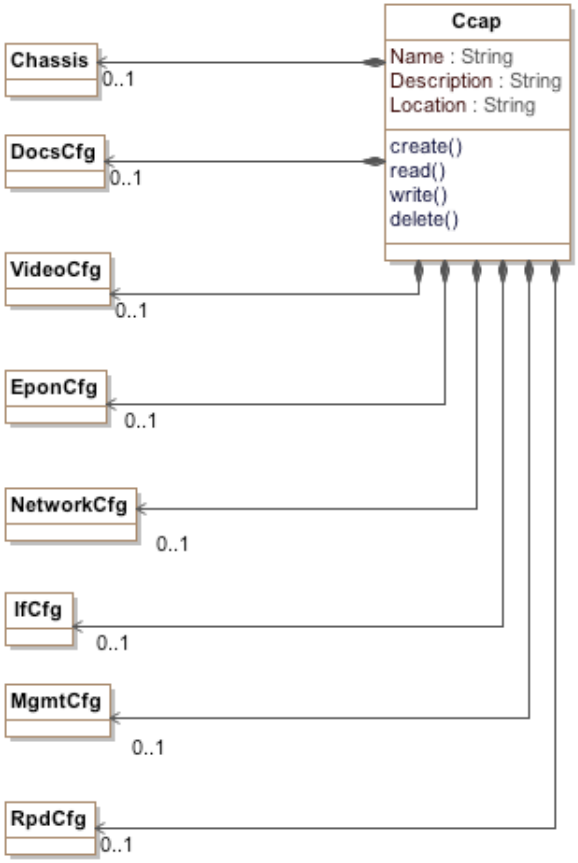

Figure 6-1 - CCAP Configuration Objects

6.4.1 Ccap Object

The Ccap object serves as the root of the CCAP configuration data. It is largely defined in [CCAP-OSSIv3.1], but is extended here via the RpdCfg object and changes beneath the Chassis object. All other objects are unmodified from [CCAP-OSSIv3.1] and are described fully there.

Table 6-1 - New Ccap Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
RpdCfg	Directed composition to RpdCfg		0..1	

6.4.1.1 RpdCfg

This configuration object is included in Figure 6-1 for reference. The RpdCfg object is defined in Section 6.4.3.

6.4.2 CCAP Chassis Objects

The Chassis configuration object model has been modified for the Remote PHY architecture; those changes are described in the following subsections. Objects not defined here are unchanged and are detailed in [CCAP-OSSIv3.1].

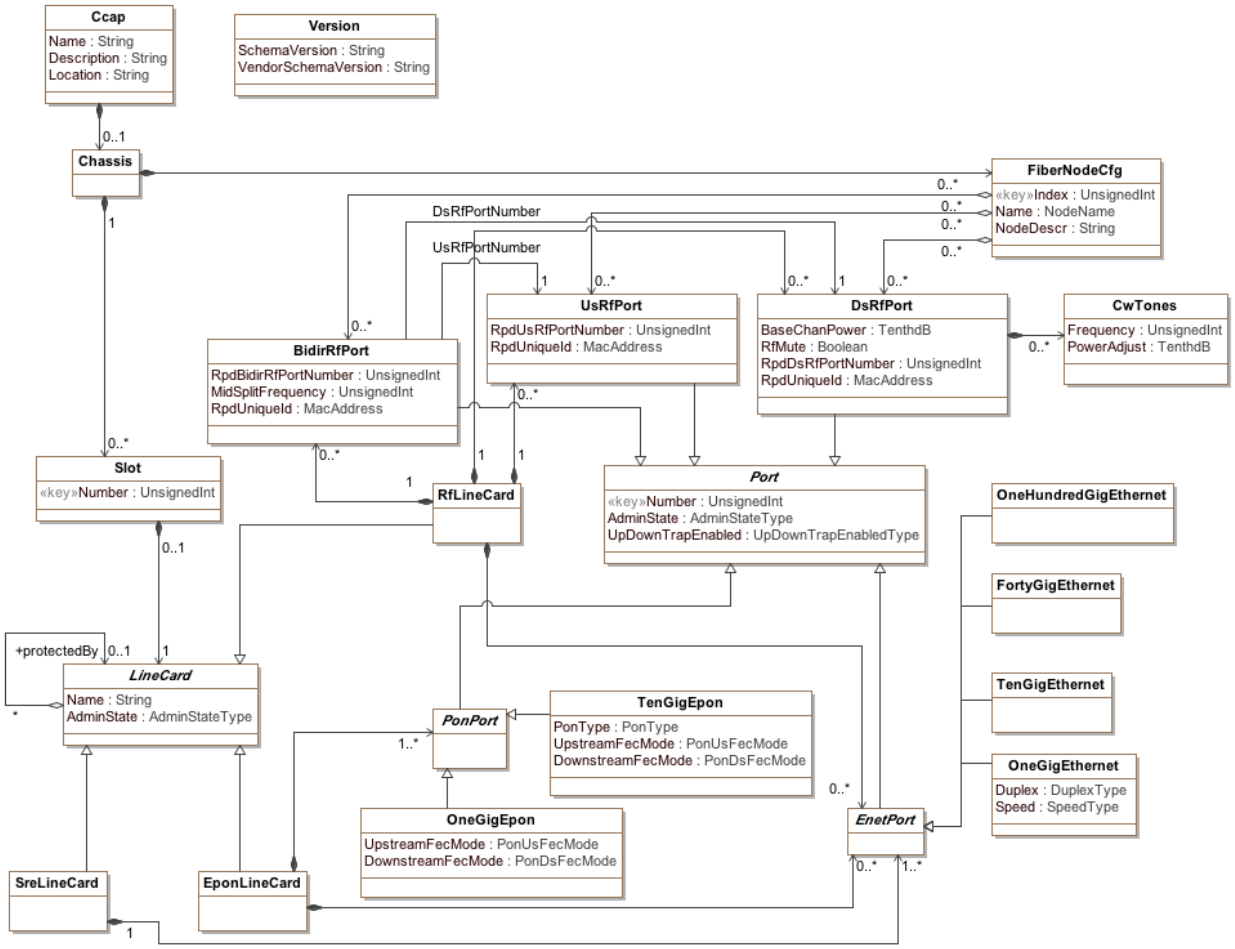

***Figure 6-2 - CCAP Chassis Objects*[4]**

[4] Modified per R-OSSI-N-16.1454-1 and R-OSSI-N-16.1469-1 on 4/27/16 by KB.

6.4.2.1 RfLineCard

This object holds the configuration data for the RF line card in a CCAP; it is extended for R-PHY to represent the RF ports on the RPD and (optionally) the Ethernet ports on the RF Line Card used to communicate with the RPD.

For R-PHY an RfLineCard can contain zero or more logical DsRfPort and UsRfPort objects which on a CCAP Core represent the configuration of physical ports on an RPD, similar to what is traditionally defined for RfLineCards in [CCAP-OSSIv3.1]. The RfLineCard can also optionally contain zero or more BidirRfPort objects, each of which represent a physical bidirectional RF port on an RPD; each BidirRfPort references the logical DsRfPort and UsRfPort objects with which it is associated.

This definition allows the flexibility for an RfLineCard to support either 1) BidirRfPorts with associated logical DsRfPorts and UsRfPorts, 2) only DsRfPorts, 3) only UsRfPorts, or 4) both DsRfPorts and UsRfPorts. Additionally, an RfLineCard can optionally support Ethernet ports used for communication between the RfLineCard and the RPD.

A BidirRfPort instance refers to a DsRfPort and a UsRfPort instance.

The new associations for the RfLineCard are listed in Table 6-2.

Table 6-2 - New RfLineCard Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
BidirRfPort	Directed composition to BidirRfPort	1	0..*	
EnetPort	Directed composition to EnetPort	1	0..*	

6.4.2.2 Port

The Port object is an abstract class from which all physical port objects on CCAP line cards are derived. There are no Port objects instantiated *per se* in an XML instance file; only the derived physical port objects are instantiated. All physical port objects that derive from Port contain the attributes of a Port which are defined in [CCAP-OSSIv3.1].

6.4.2.3 DsRfPort

This object allows for the configuration of a physical Downstream RF port on an RfLineCard or a logical Downstream RF port whose physical port is located on an RPD. For Remote PHY, a DsRfPort object also contains the new attributes in the following table. Other attributes are unchanged from [CCAP-OSSIv3.1]. Note, however, that the Tilt and MaxTiltFrequency attributes generally only apply to a Remote PHY device deployed in a headend or hub location where the downstream signals enter the combining network and are fed to traditional analog lasers. These attributes are not configured for an RPD with a flat output, but could be configured if the RPD supports tilted output.

Table 6-3 - New DsRfPort Object Attributes[5]

Attribute Name	Type	Required Attribute	Type Constraints	Units	Default Value
RpdDsRfPortNumber	UnsignedInt	No			
RpdUniqueId	MacAddress	No			

The DsRfPort has the following new associations.

[5] Modified per R-OSSI-N-16.1469-1 on 4/25/16 b KB.

Table 6-4 - DsRfPort Object Associations[6]

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
CwTones	Directed composition to CwTones	1	0..*	

6.4.2.3.1 New DsRfPort Object Attributes

6.4.2.3.1.1 RpdDsRfPortNumber

This attribute identifies the physical port on the RPD that this logical DS RF port represents.

This attribute is omitted if this logical DsRfPort is associated with a physical BidirRfPort on the RPD.

6.4.2.3.1.2 RpdUniqueId[7]

This attribute configures the IP address (v4 or v6) or FQDN with which an RPD originates its GCP connection to this CCAP Core.

This attribute is omitted if this logical DsRfPort is associated with a physical BidirRfPort on the RPD.

6.4.2.4 CwTones[8]

This new R-PHY object allows CW tones to be configured on a DS RF port of a CCAP Core associated with an RPD. The CW tone is commonly used on downstream RF ports for automatic gain control (pilot tone) as an alignment carrier or for other purposes. Multiple CW tones can be configured for a downstream RF port.

Table 6-5 - CwTones Object Attributes

Attribute Name	Type	Required Attribute	Type Constraints	Units	Default Value
Frequency	UnsignedInt	Yes		Hz	
PowerAdjust	UnsignedByte	Yes, see description	TenthdB		

6.4.2.4.1 CwTones Object Attributes

6.4.2.4.1.1 Frequency

This attribute is the RF frequency of this CW tone.

6.4.2.4.1.2 PowerAdjust

This attribute represents the power gain for the CW tone relative to the BaseChanPower for this DsRfPort. It is expressed in TenthdB.

6.4.2.5 UsRfPort

A UsRfPort object represents a physical Upstream RF port on an RfLineCard or a logical Upstream RF port whose physical port is located on an RPD. For Remote PHY, an UsRfPort object contains the new attributes in the following table.

[6] Modified per R-OSSI-N-16.1454-1 on 4/25/16 by KB.
[7] Modified per R-OSSI-N-16.1469-1 on 4/25/16 b KB.
[8] Modified per R-OSSI-N-16.1454-1 on 4/25/16 by KB.

Table 6-6 - New UsRfPort Object Attributes[9]

Attribute Name	Type	Required Attribute	Type Constraints	Units	Default Value
RpdUsRfPortNumber	UnsignedInt	No			
RpdUniqueId	MacAddress	No			

A UsRfPort's associations are defined in [CCAP-OSSIv3.1].

6.4.2.5.1 New UsRfPort Object Attributes

6.4.2.5.1.1 RpdUsRfPortNumber

This attribute identifies the physical port on the RPD that this logical US RF port represents.

This attribute is omitted if this logical UsRfPort is associated with a physical BidirRfPort on the RPD.

6.4.2.5.1.2 RpdUniqueId[10]

This attribute configures the IP address (v4 or v6) or FQDN with which an RPD originates its GCP connection to this CCAP Core.

This attribute is omitted if this logical UsRfPort is associated with a physical BidirRfPort on the RPD.

6.4.2.6 BidirRfPort

This new R-PHY object corresponds to a physical bidirectional RF port on an RPD. This object allows for the configuration of the association between a physical bidirectional RF port on an RPD and the logical DsRfPorts and UsRfPorts (from the CCAP's perspective) that hold the downstream and upstream configuration of that bidirectional port. The BidirRfPort is a type of the abstract class Port and inherits those common parameters. A BidirRfPort object contains the attributes in the following table.

Table 6-7 - BidirRfPort Object Attributes[11]

Attribute Name	Type	Required Attribute	Type Constraints	Units	Default Value
RpdBidirRfPortNumber	UnsignedInt	Yes			
MidSplitFrequency	UnsignedInt	No		Hz	
RpdUniqueId	MacAddress	Yes			

Table 6-8 - BidirRfPort Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
Port	Specialization of Port			
DsRfPort	Directed association to DsRfPort	1	1	DsRfPortNumber
UsRfPort	Directed association to UsRfPort	1	1	UsRfPortNumber

6.4.2.6.1 BidirRfPort Object Attributes

6.4.2.6.1.1 RpdBidiRfPortNumber

This attribute is the port number of this Bidirectional RF Port from the RPD's Bidirectional RF port number space.

[9] Modified per R-OSSI-N-16.1469-1 on 4/25/16 by KB.
[10] Modified per R-OSSI-N-16.1469-1 on 4/25/16 by KB.
[11] Modified per R-OSSI-N-16.1469-1 on 4/25/16 by KB.

6.4.2.6.1.2 MidSplitFrequency

This attribute is the RF frequency of the midsplit in the cable system connected to this Bidirectional RF Port. This attribute can be omitted if it is not configurable in the RPD.

6.4.2.6.1.3 RpdUniqueId[12]

This attribute configures the IP address (v4 or v6) or FQDN with which an RPD originates its GCP connection to this CCAP Core.

6.4.2.7 FiberNodeCfg

The FiberNodeCfg object defines the cable hybrid fiber/coax system (HFC) plant Fiber Nodes reached by RF ports on a CCAP. FiberNode attributes are defined in [CCAP-OSSIv3.1].

The FiberNodeCfg object has the following associations.

Table 6-9 - New FiberNodeCfg Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
BidirRfPort	Directed aggregation to BidirRfPort	0..*	0..*	BidirRfPort

When using BidirRfPorts, the CCAP SHOULD reject a configuration where a FiberNodeCfg contains a DsRfPort or UsRfPort which are associated with a BidirRfPort. This is because BidirRfPorts already contain references to UsRfPorts and DsRfPorts.

6.4.3 RpdCfg Objects

The RPD configuration objects are new for the Remote PHY architecture.

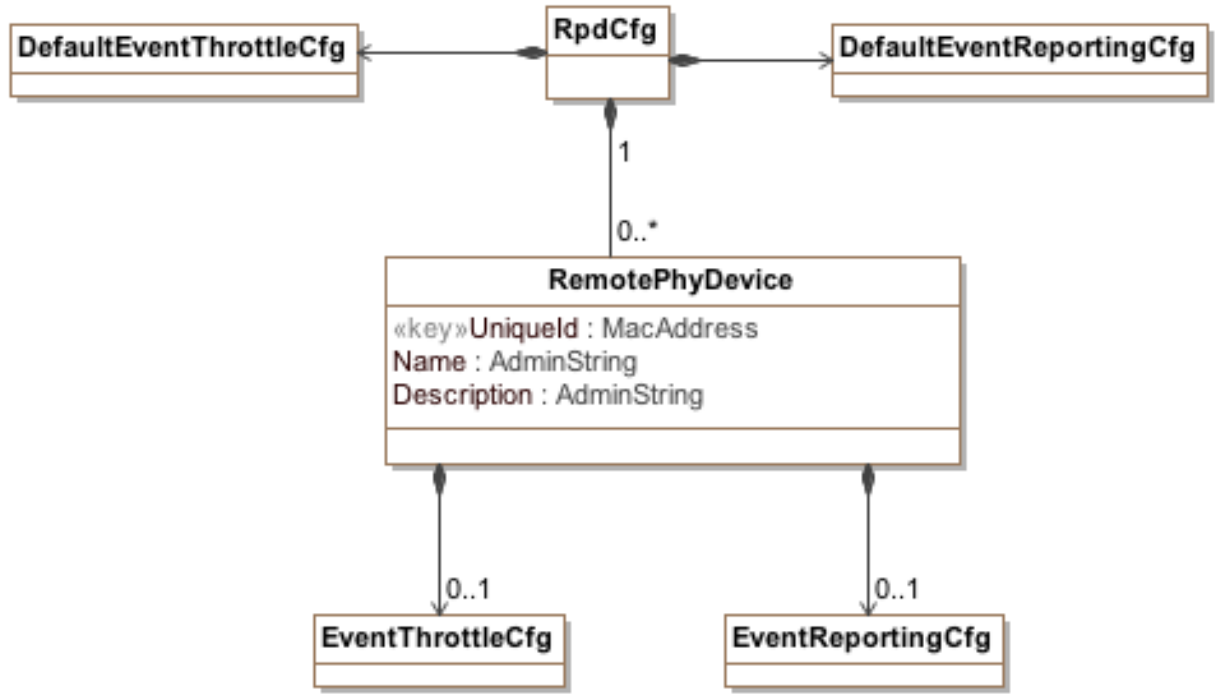

Figure 6-3 - CCAP Rpd Objects[13]

[12] Modified per R-OSSI-N-16.1469-1 on 4/25/16 b KB.

6.4.3.1 RpdCfg

The RpdCfg object is a container that holds RemotePhyDevice instances and has the associations shown in Table 6-10.

Table 6-10 - RpdCfg Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
RemotePhyDevice	Directed composition to RemotePhyDevice	1	0..*	
DefaultEventThrottleCfg	Directed composition to DefaultEventThrottleCfg	1	1	
DefaultEventReportingCfg	Directed composition to DefaultEventReportingCfg	1	1	

6.4.3.2 DefaultEventThrottleCfg

This object configures the default event throttling parameters for RPDs. If an instance of RemotePhyDevice is configured without an EventThrottleCfg object, the configuration here applies. It is based on the EventThrottleCfg object defined in [CCAP-OSSIv3.1] and is used here without modification.

6.4.3.3 DefaultEventReportingCfg

This object configures the default event reporting parameters for RPDs. If an instance of RemotePhyDevice is configured without an EventReportingCfg object, the configuration here applies. It is based on the EventReportingCfg object defined in [CCAP-OSSIv3.1] and is used here without modification.

6.4.3.4 RemotePhyDevice

The RemotePhyDevice object allows the user to optionally configure attributes of the Remote PHY Devices for reporting purposes.

Table 6-11 - RemotePhyDevice Object Attributes[14]

Attribute Name	Type	Required Attribute	Type Constraints	Units	Default Value
UniqueId	MacAddress	Yes (key)			
Name	AdminString	No			
Description	AdminString	No			

Table 6-12 - RemotePhyDevice Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
EventThrottleCfg	Directed composition to EventThrottleCfg	1	0..1	
EventReportingCfg	Directed composition to EventReportingCfg	1	0..1	

6.4.3.4.1 RemotePhyDevice Object Attributes

6.4.3.4.1.1 UniqueId[15]

This attribute configures the IP address (v4 or v6) or FQDN with which an RPD originates GCP to this CCAP Core.

[13] Modified per R-OSSI-N-16.1469-1 on 4/27/16 by KB.
[14] Modified per R-OSSI-N-16.1469-1 on 4/25/16 b KB.
[15] Modified per R-OSSI-N-16.1469-1 on 4/25/16 b KB.

6.4.3.4.1.2 Name

This attribute configures a short name of the RemotePhyDevice for reporting. While not a key, the Name of the RemotePhyDevice is required to be unique on this CCAP.

6.4.3.4.1.3 Description

This attribute configures an informational description of the RemotePhyDevice.

6.4.3.5 EventThrottleCfg

This object configures specific event throttling parameters for a RemotePhyDevice instance. It is defined in [CCAP-OSSIv3.1].

6.4.3.6 EventReportingCfg

This object configures specific event reporting parameters for a RemotePhyDevice instance. It is defined in [CCAP-OSSIv3.1].

6.4.4 Downstream RF Port Configuration Objects

The downstream RF port configuration object model has been modified for the Remote PHY architecture; those changes are described in the following subsections. Objects not defined here are unchanged and are detailed in [CCAP-OSSIv3.1].

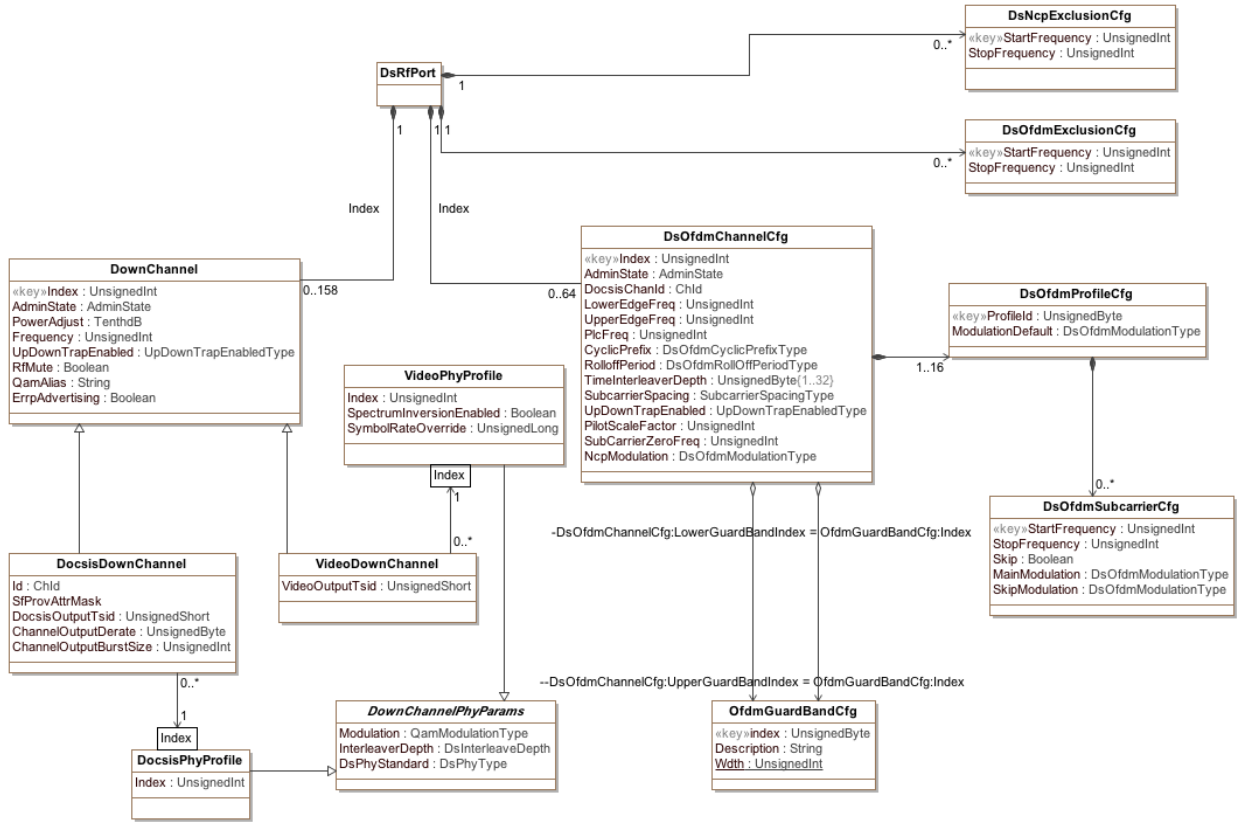

Figure 6-4 - CCAP Downstream RF Port Configuration Objects

6.4.4.1 Downstream RF Port

The DsRfPort object allows the user to configure the CCAP Downstream RF Ports elements either on an integrated CCAP or CCAP connected to a Remote PHY Device. The DsRfPort object has the following objects which are modified by this specification; all other objects and associations are as defined in [CCAP-OSSIv3.1].

6.4.4.1.1 DocsisDownChannel

The DocsisDownChannel object is a DownChannel used exclusively for DOCSIS. Two attributes have been added to the DocsisDownChannel object for Remote PHY support: ChannelOutputDerate and ChannelOutputBurstSize. Otherwise the DocsisDownChannelObject is unmodified from its definition in [CCAP-OSSIv3.1].

Table 6-13 - New DocsisDownChannel Object Attributes

Attribute Name	Type	Required Attribute	Type Constraints	Units	Default Value
ChannelOutputDerate	UnsignedByte	No	90..100	%	99%
ChannelOutputBurstSize	UnsignedInt	No		Bytes	

6.4.4.1.1.1 DocsisDownChannel Object Attributes

6.4.4.1.1.1.1 ChannelOutputDerate

The percentage of the maximum output rate for the aggregated traffic that is being sent though this Downstream interface to the Downstream channel associated with this DEPI session. Using a value lower than 100% of the Downstream channel's configured payload rate prevents the buildup of a queue delay when MPEG-TS nulls are added in the presence of jitter in the CIN.

6.4.4.1.1.1.2 ChannelOutputBurstSize

The maximum burst size for the aggregate output rate of traffic that is being sent through this Downstream interface to the Downstream channel. The default value of this object corresponds to 3 CCAP Core payload MTUs.

6.5 RPD Control Objects[16]

These objects allow direct control of different aspects of a specific RPD. The RPD MUST implement the RPD control objects specified in Figure 6-5. Implementation is vendor-specific and can be accomplished via mechanisms such as a command line interface.

[16] Added per R-OSSI-N-15.1378-1 on 11/12/15 by KB.

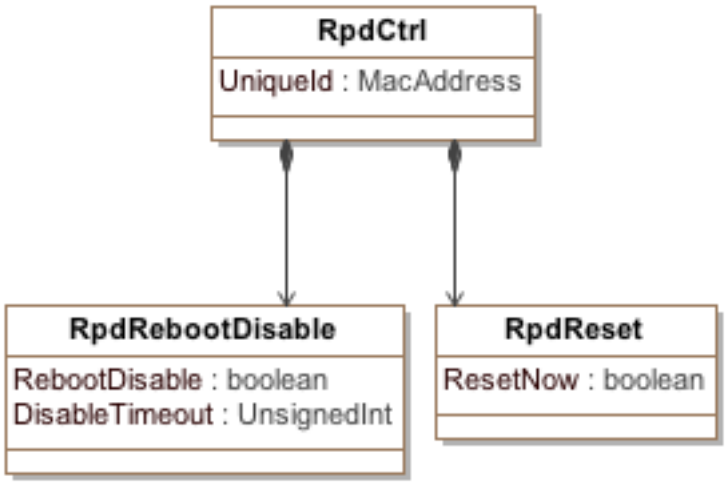

Figure 6-5 - RPD Control Objects[17]

6.5.1 RpdCtrl

The RpdCtrl object is the primary container of RPD Control objects. It has the following associations:

Table 6-14 - RpdCtrl Object Attributes[18]

Attribute Name	Type	Required Attribute	Type Constraints	Units	Default Value
UniqueId	MacAddress	Yes (Key)			

Table 6-15 - RpdCtrl Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
RpdReset	Directed composition to RpdReset			
RpdRebootDisable	Directed composition to RpdRebootDisable			

6.5.1.1 UniqueId[19]

This attribute specifies the IP address (v4 or v6) or FQDN of the RPD to which the command is being sent.

6.5.2 RpdReset

This control object allows an RPD to be reset remotely.

Table 6-16 - RpdReset Object Attributes

Attribute Name	Type	Required Attribute	Type Constraints	Units	Default Value
ResetNow	Boolean	No			false

[17] Modified per R-OSSI-N-16.1469-1 on 4/27/16 by KB.
[18] Modified per R-OSSI-N-16.1469-1 on 4/25/16 by KB.
[19] Modified per R-OSSI-N-16.1469-1 on 4/25/16 by KB.

6.5.2.1 ResetNow

This attribute controls whether or not the RPD begins a reboot. When set to true, the RPD reboots. This value resets to false on reinitialization.

6.5.3 RpdRebootDisable

This control object disables automatic reboot of the RPD for a specified period of time to allow remote connection to an RPD that is uninterrupted by an automatic reboot. The RPD reboots automatically if it is not able to successfully complete the initialization process defined in [R-PHY]. If an RPD is unable to initialize and is stuck in a cycle of automatic reboots, this object allows the automatic reboot to be disabled so that the debugging process is not interrupted by automatic RPD reboot. The reboot disable automatically times out so that an RPD is not accidentally kept from rebooting in the future.

Table 6-17 - RpdRebootDisable Object Attributes

Attribute Name	Type	Required Attribute	Type Constraints	Units	Default Value
RebootDisable	Boolean	No			false
DisableTimeout	UnsignedInt	No	1..3600	Seconds	360

6.5.3.1 RebootDisable

The value of this attribute sets whether the automatic reboot should be delayed or not. If set to true, the RPD will not reboot until the value in the DisableTimeout attribute has elapsed. If reboot has been disabled, setting this value to false allows the RPD to automatically reboot. If the RPD had failed to initialize, it could reboot when the value is set to false. This value resets to false on reinitialization.

6.5.3.2 DisableTimeout

When RebootDisable is set to true, this attribute controls how long the RPD should wait until it reboots and begins the initialization process again. The timer countdown begins when RebootDisable is set to true. This value resets to the default on reinitialization.

6.6 CCAP Core RPD Secure Software Download Control Objects[20]

The CCAP-Core is responsible for facilitating secure updates to software running on RPDs. The RPD Secure Software Download process is described in the Secure Software Download section of [R-PHY].

6.6.1 CCAP-Core Objects for RPD SSD Management

Table 6-18 summarizes the CCAP-Core objects used for RPD SSD management. These objects can be written by an external NMS to initiate RPD SSD process of SSD.

Table 6-18 - CCAP-Core Objects for RPD SSD Management[21]

Attribute Name	Type	Access	Type Constraints	Units
RpdId	MacAddress	N/A	Key	
SsdServerAddress	IpAddress	R/W		
SsdTransport	unsignedByte	R/W	TFTP(1), HTTP(2)	
SsdFilename	String	R/W		
SsdManufCvcChain	Hexstring	R/W		
SsdCosignerCvcChain	Hexstring	R/W		
SsdAdminControl	unsignedByte	R/W	Disable (1), Enable (2),	

[20] Added per R-OSSI-N-15.1409-2 on 2/3/15 by KB.
[21] Table modified per R-OSSI-N-16.1475-2 on 4/27/16 by KB.

Attribute Name	Type	Access	Type Constraints	Units
SsdStatus	unsignedByte	R/O	Idle(1), SsdReady (2), SsdStarted (3), SsdCompleted (4), SsdFailed (5)	

6.6.1.1 RpdId

The RpdId object identifies the RPD for which the SSD needs to be performed.

6.6.1.2 SsdServerAddress

The SsdServerAddress object identifies the SSD server in the form of IPv4 or IPv6 address.

6.6.1.3 SsdTransport

The SsdTransport object communicates the type of transport for the RPD download of the software file. The defined values are:

1 - TFTP

2 - HTTP

All other values are reserved.

6.6.1.4 SsdFilename

The SsdFilename object is used to communicate the name of the software file which the RPD needs to download.

6.6.1.5 SsdManufCvcChain

This object is used to communicate the certificate chain from the new PKI that contains both the Manufacturer Code Verification Certificate and the certification authority (CA) certificate that issued the Manufacturer Code Verification Certificate for Secure Software Download. The Manufacturer CVC Chain TLV (M-CVC-C) is used to enable the RPD to download the code file from the download server.

6.6.1.6 SsdCosignerCvcChain

This object is used to communicate the certificate chain from the new PKI that contains both the Co-signer Code Verification Certificate and the certification authority (CA) certificate that issued the Co-signer Code Verification Certificate for Secure Software Download. The Co-signer CVC Chain TLV (C-CVC-C) is used to enable the RPD to download the code file from the download server.

6.6.1.7 SsdAdminControl

The SsdAdminControl object allows enabling or disabling SSD for a selected RPD.

The defined values are:

1 - Disable

2 - Enable.

6.6.1.8 SsdStatus

The CCAP Core reports the status of SSD process through SsdStatus object.

The defined values are:

1 - Idle

2 - SsdReady

3 - SsdStarted

4 - SsdCompleted

5- SsdFailed

7 PERFORMANCE MANAGEMENT[22]

This section defines CCAP Core and RPD requirements for performance management functions.

7.1 Performance Management UML Object Model

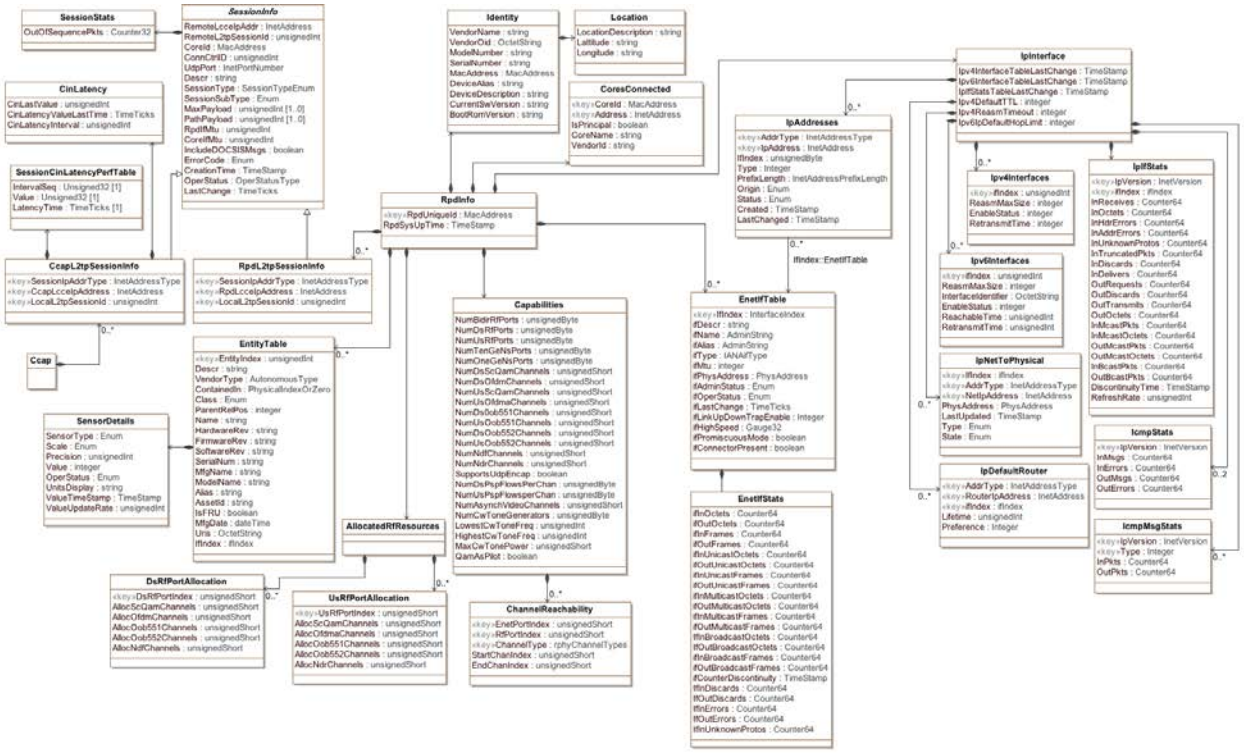

Figure 7-1 - DOCS-RPHY-MIB Information Model[23]

7.1.1 RpdInfo[24]

This object identifies the RPD for which the details and statistics are being provided.

Table 7-1 - RpdInfo Object[25]

Attribute Name	Type	Access	Type Constraints	Units
RpdUniqueId	MacAddress	key		N/A
RpdSysUpTime	TimeStamp	read-only		

Table 7-2 - RpdInfo Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
Identity	Directed composition to Identity	1	1	
Capabilities	Directed composition to Capabilities	1	1	

[22] Modified per R-OSSI-N-16.1443-2 and R-OSSI-N-16.1469-1on 4/27/16 by KB.
[23] Modified per R-OSSI-N-16.1443-2 and R-OSSI-N-16.1469-1on 4/27/16 by KB.
[24] This section and subsections modified per R-OSSI-N-16.1443-2 on 4/22/16 by KB.
[25] Modified per R-OSSI-N-16.1469-1 on 4/25/16 by KB.

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
CoresConnected	Directed composition to CoresConnected	1	1	
AllocatedResources	Directed composition to AllocatedResources	1	1	
RpdL2tpSessionInfo	Directed composition to RpdL2tpSessionInfo	1	0..*	
PwDetails	Directed composition to PwDetails	1	0..*	
EnetIfTable	Directed composition to EnetIfTable	1	0..*	
IpInterface	Directed composition to IpAddress	1	1	
EntityTable	Directed composition to EntityTable	1	0..*	
Sensors	Directed composition to Sensors	1	0..*	

7.1.1.1 RpdUniqueId[26]

This attribute specifies the MAC address associated with the lowest numbered CIN facing Ethernet port.

7.1.1.2 RpdSysUpTime

The time (in hundredths of a second) since the RPD was last re-initialized. This value is reported by the RPD.

7.1.2 Identity

This object provides data that uniquely identifies the RPD.

Table 7-3 - Identity Object

Attribute Name	Type	Access	Type Constraints	Units
VendorName	string	Read-only		N/A
VendorId	unsignedShort	Read-only		N/A
ModelNumber	string	Read-only		N/A
SerialNumber	string	Read-only		N/A
MacAddress	MacAddress	Read-only		N/A
DeviceAlias	string	Read-only		N/A
DeviceDescription	string	Read-only		N/A
CurrentSwVersion	string	Read-only		N/A
BootRomVersion	string	Read-only		N/A

Table 7-4 - Identity Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
Location	Directed composition to Location	1	1	

7.1.2.1 VendorName

Identifies the RPD manufacturer. The format is vendor proprietary.

7.1.2.2 VendorId

The IANA-assigned SMI Network Management Private Enterprise Code of the vendor, as specified in [RFC 5612].

[26] Modified per R-OSSI-N-16.1469-1 on 4/25/16 by KB.

7.1.2.3 ModelNumber

The model name and number used by the vendor to identify the RPD. The format is vendor proprietary.

7.1.2.4 SerialNumber

The serial number of the RPD. The format is vendor proprietary.

7.1.2.5 MacAddress

The main MAC address of the RPD. Typically, the MAC address associated with the lowest numbered CIN facing Ethernet port.

7.1.2.6 DeviceAlias

A device name assigned by the operator via the management interface. The object provides a non-volatile "handle" for the RPD.

7.1.2.7 DeviceDescription

A short text description of the RPD provided by the RPD manufacturer.

7.1.2.8 CurrentSWVersion

The version number of the software currently running on the RPD. The format is vendor proprietary.

7.1.2.9 BootRomVersion

The version number of the boot ROM currently installed on the RPD. The format is vendor proprietary.

7.1.3 Location

This object provides location details for the RPD. The values are populated via a management interface or other mechanisms.

Table 7-5 - Location Object

Attribute Name	Type	Access	Type Constraints	Units
LocationDescription	string	Read-only		N/A
Latitude	string	Read-only		N/A
Longitude	string	Read-only		N/A

7.1.3.1 LocationDescription

A short text description of where the RPD has been installed, such as a street address. The format is specific to the operator.

7.1.3.2 Latitude

The latitudinal coordinate of the RPD location, expressed as a 6-byte long string as described in [ISO 6709] (6 digit degrees, minutes, seconds: ±DDMMSS.S). For example: -750015.1.

This value could be provided by a GPS receiver within the module.

7.1.3.3 Longitude

The longitudinal coordinate of the RPD location, expressed as a 7-byte long string as described in [ISO 6709] (7 digits degrees, minutes, seconds: ±DDDMMSS.S). For example: -0750015.1.

This value could be provided by a GPS receiver within the module.

7.1.4 CoresConnected

This object provides a list of CCAP Cores to which the RPD is authenticated, including the CCAP Core on which the MIB is polled. For each CCAP Core entry, the PRD indicates if that CCAP Core is the principal Core. These values are provided by the CCAP Core on initialization.

Table 7-6 - CoresConnected Object[27]

Attribute Name	Type	Access	Type Constraints	Units
CoreId	MacAddress	key		N/A
Address	InetAddress	Read-only		N/A
IsPrincipal	boolean	Read-only		N/A
CoreName	string	Read-only		N/A
VendorId	unsignedShort	Read-only		N/A

7.1.4.1 CoreId

Provides the MAC address of the CCAP Core identified in the row entry and acts as a key.

7.1.4.2 Address

Provides the IPv4 or IPv6 address of the CCAP Core.

7.1.4.3 IsPrincipal

If true, indicates that this CCAP Core is the principal Core.

7.1.4.4 CoreName

Provides the name of the CCAP Core as conveyed to the RPD.

7.1.4.5 VendorId

Provides the IANA-assigned SMI Network Management Private Enterprise Code of the vendor, as specified in [RFC 5612].

7.1.5 Capabilities

This object provides information about the principal capabilities and constraints of the RPD.

Table 7-7 - Capabilities Object

Attribute Name	Type	Access	Type Constraints	Units
NumBidirRfPorts	unsignedByte	Read-only		N/A
NumDsRfPorts	unsignedByte	Read-only		N/A
NumUsRfPorts	unsignedByte	Read-only		N/A
NumTenGeNsPorts	unsignedByte	Read-only		N/A
NumOneGeNsPorts	unsignedByte	Read-only		N/A
NumDsScQamChannels	unsignedShort	Read-only		N/A
NumDsOfdmChannels	unsignedShort	Read-only		N/A
NumUsScQamChannels	unsignedShort	Read-only		N/A
NumUsOfdmaChannels	unsignedShort	Read-only		N/A
NumDsOob551Channels	unsignedShort	Read-only		N/A
NumUsOob551Channels	unsignedShort	Read-only		N/A

[27] Modified per R-OSSI-N-16.1469-1 on 4/27/16 by KB.

Attribute Name	Type	Access	Type Constraints	Units
NumDsOob552Channels	unsignedShort	Read-only		N/A
NumUsOob552Channels	unsignedShort	Read-only		N/A
NumNdfChannels	unsignedShort	Read-only		N/A
NumNdrChannels	unsignedShort	Read-only		N/A
SupportsUdpEncap	boolean	Read-only		N/A
NumDsPspFlowsPerChan	unsignedByte	Read-only		N/A
NumUsPspFlowsPerChan	unsignedByte	Read-only		N/A
NumAsynchVideoChannels	unsignedShort	Read-only		N/A
NumCwToneGenerators	unsignedByte	Read-only		N/A
LowestCwToneFreq	unsignedInt	Read-only		Hz
HighestCwToneFreq	unsignedInt	Read-only		Hz
MaxCwTonePower	unsignedShort	Read-only		TenthdBmV
QamAsPilot	boolean	Read-only		N/A

Table 7-8 - Capabilities Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
ChannelReachability	Directed composition to ChannelReachability	1	0..*	

7.1.5.1 NumBidirRfPorts

Provides the number of bidirectional RF ports available on the RPD.

7.1.5.2 NumDsRfPorts

Provides the number of downstream unidirectional RF ports available on the RPD.

7.1.5.3 NumUsRfPorts

Provides the number of upstream unidirectional RF ports available on the RPD.

7.1.5.4 NumTenGeNsPorts

Provides the number of 10 gigabit Ethernet ports supported by the RPD.

7.1.5.4.1 NumOneGeNsPorts

Provides the number of 1 gigabit Ethernet ports supported by the RPD.

7.1.5.4.2 NumDsScQamChannels

Provides the number of downstream SC-QAM channels supported per downstream RF port.

7.1.5.4.3 NumDsOfdmChannels

Provides the number of downstream DOCSIS 3.1 channels supported per downstream RF port.

7.1.5.4.4 NumUsScQamChannels

Provides the number of upstream SC-QAM channels supported per upstream RF port.

7.1.5.4.5 NumUsOfdmaChannels

Provides the number of upstream DOCSIS 3.1 channels supported per upstream RF port.

7.1.5.4.6 NumDsOob551Channels

Provides the number of downstream SCTE 55-1 channels supported per downstream RF port.

7.1.5.4.7 NumUsOob551Channels

Provides the number of upstream SCTE 55-1 channels supported per upstream RF port.

7.1.5.4.8 NumDsOob552Channels

Provides the number of downstream SCTE 55-2 channels supported per downstream RF port.

7.1.5.4.9 NumUsOob552Channels

Provides the number of upstream SCTE 55-2 channels supported per upstream RF port.

7.1.5.4.10 NumNdfChannels

Provides the number of narrowband digital forward channels supported per downstream RF port.

7.1.5.4.11 NumNdrChannels

Provides the number of narrowband digital return channels supported per upstream RF port.

7.1.5.4.12 SupportsUdpEncap

If true, indicates that the RPD supports UDP encapsulation on L2TPv3 pseudowires.

7.1.5.4.13 NumDsPspFlows

Provides the number of distinct PSP flows supported by the RPD on downstream data pseudowires.

7.1.5.4.14 NumUsPspFlows

Provides the number of distinct PSP flows supported by the RPD on upstream data pseudowires.

7.1.5.4.15 NumAsyncVideoChannels

Provides the number of asynchronous MPEG video channels supported per downstream RF port.

7.1.5.4.16 NumCwToneGens

Provides the number of CW tone generators supported per downstream RF port.

7.1.5.4.17 LowestCwToneFreq

Provides the lowest frequency supported by the CW tone generators.

7.1.5.4.18 HighestCwToneFreq

Provides the highest frequency supported by the CW tone generators.

7.1.5.4.19 MaxCwTonePower

Provides the maximum power level supported by the dedicated CW tone generators, expressed in TenthdBmV.

7.1.5.4.20 QamAsPilot

If true, indicates that a QAM channel can be configured as a CW tone.

7.1.6 ChannelReachability

In some RPD implementations, an Ethernet interface might not have connectivity to all channels on a port of the RPD. This object allows the RPD to communicate those constraints. This table is only populated if reachability constraints exist on the RPD.

Table 7-9 - ChannelReachability Object

Attribute Name	Type	Access	Type Constraints	Units
EnetPortIndex	unsignedShort	key		N/A
RfPortIndex	unsignedShort	key		N/A
ChannelType	enum	key	dsScQam(1), dsOfdm(2), dsOob551(3), dsOob552(4), ndf(5), usScQam(6), usOfdma(7), usOob551(8), usOob552(9), ndr(10)	N/A
StartChanIndex	unsignedShort	Read-only		N/A
EndChanIndex	unsignedShort	Read-only		N/A

7.1.6.1 EnetPortIndex

Identifies the Ethernet port on the RPD that has the connectivity constraint.

7.1.6.2 RfPortIndex

Identifies the RF port with which the Ethernet port has a connectivity constraint.

7.1.6.3 ChannelType

Identifies the type of channel that is supported within the specified channel index range on this RF port from the specified Ethernet interface. A row entry will be created for each channel type with a constraint. Absence of a row for a channel type means there is no constraint for that channel type.

7.1.6.4 StartChanIndex

Identifies the first channel of the specified channel type in the range of channels that does not have connectivity to the specified Ethernet port.

7.1.6.5 EndChanIndex

Identifies the last channel of the specified channel type in the range of channels that does not have connectivity to the specified Ethernet port.

7.1.7 AllocatedRfResources

Provides the allocation status for downstream and upstream channel resources on the RPD on a per-port basis.

Table 7-10 - AllocatedRfResources Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
DsRfPortAllocation	Directed composition to DsRfPortAllocation	1	0..*	
UsRfPortAllocation	Directed composition to UsRfPortAllocation	1	0..*	

7.1.8 DsRfPortAllocation

Provides the allocation status for downstream channel resources on the RPD on a per DS RF port basis.

Table 7-11 - DsRfPortAllocation Object

Attribute Name	Type	Access	Type Constraints	Units
DsRfPortIndex	unsignedShort	key		N/A
AllocScQamChannels	unsignedShort	Read-only		N/A
AllocOfdmChannels	unsignedShort	Read-only		N/A
AllocOob551Channels	unsignedShort	Read-only		N/A
AllocOob552Channels	unsignedShort	Read-only		N/A
AllocNdfChannels	unsignedShort	Read-only		N/A

7.1.8.1 DsRfPortIndex

Provides the index of the downstream RF port for which resource allocation is being reported.

7.1.8.2 AllocScQamChannels

Provides the number of allocated SC-QAM channels on this RF port.

7.1.8.2.1 AllocOfdmChannels

Provides the number of allocated DOCSIS 3.1 channels on this RF port.

7.1.8.2.2 AllocOob551Channels

Provides the number of allocated SCTE 55-1 channels on this RF port.

7.1.8.2.3 AllocOob552Channels

Provides the number of allocated SCTE 55-2 channels on this RF port.

7.1.8.2.4 AllocNdfChannels

Provides the number of allocated narrowband digital forward channels on this RF port.

7.1.9 UsRfPortAllocation

Provides the allocation status for upstream channel resources on the RPD on a per US RF port basis.

Table 7-12 - UsRfPortAllocation Object

Attribute Name	Type	Access	Type Constraints	Units
UsRfPortIndex	unsignedShort	key		N/A
AllocScQamChannels	unsignedShort	Read-only		N/A
AllocOfdmaChannels	unsignedShort	Read-only		N/A
AllocOob551Channels	unsignedShort	Read-only		N/A
AllocOob552Channels	unsignedShort	Read-only		N/A
AllocNdrChannels	unsignedShort	Read-only		N/A

7.1.9.1 UsRfPortIndex

Provides the index of the upstream RF port for which resource allocation is being reported.

7.1.9.2 AllocScQamChannels

Provides the number of allocated SC-QAM channels on this RF port.

7.1.9.2.1 AllocOfdmaChannels

Provides the number of allocated DOCSIS 3.1 channels on this RF port.

7.1.9.2.2 AllocOob551Channels

Provides the number of allocated SCTE 55-1 channels on this RF port.

7.1.9.2.3 AllocOob552Channels

Provides the number of allocated SCTE 55-2 channels on this RF port.

7.1.9.2.4 AllocNdrChannels

Provides the number of allocated narrowband digital return channels on this RF port.

7.1.10 RpdL2tpSessionInfo[28]

The RpdL2tpSessionInfo object provides information about each tunnel session between the RPD and each CCAP Core with which it is associated from the RPD's point of view. A row entry is created for every session the RPD terminates. There may be entries for sessions with different CCAP Cores if the RPD is connected with one or more auxiliary cores.

The RpdL2tpSessionInfo object inherits the attributes of the SessionInfo object.

The CCAP Core MUST populate the RpdL2tpSessionInfo table and SessionStats table with each DEPI, UEPI, OOB, NDF, and NDR pseudowire that is established on each RPD with which it is associated.

If the RDP supports SNMP, the RPD SHOULD populate the RpdL2tpSessionInfo table with each DEPI, UEPI, OOB, NDF, and NDR pseudowire it has established with CCAP Cores.

In the case where there are multiple CCAP Cores, all sessions are reported, regardless of the CCAP Core fulfilling the MIB request.

If the RPD supports SNMP, it creates an entry in this table for each L2TPv3 tunnel (session) established with each CCAP Core to which it is connected.

Table 7-13 - RpdL2tpSessionInfo Object

Attribute Name	Type	Access	Type Constraints	Units
SessionIpAddrType	InetAddressType	Key		N/A
RpdLcceIpAddress	InetAddress	Key		N/A
LocalL2tpSessionId	UnsignedInt	Key		N/A

Table 7-14 - RpdL2tpSessionInfo Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
SessionInfo	Specialization of SessionInfo			

7.1.10.1 SessionIpAddrType

Indicates whether the IP address provide in the RpdLcceIpAddress and RemoteLcceIpAddr attributes are IPv4 or IPv6.

[28] Modified per R-OSSI-N-16.1443-2 on 4/22/16 by KB.

7.1.10.2 RpdLcceIpAddress

Provides the local LCCE IP address on the RPD of the session detailed in the row entry.

7.1.10.3 LocalL2tpSessionId

Provides the value of the session ID assigned to the session by the RPD.

7.1.11 CcapL2tpSessionInfo[29]

The CcapL2tpSessionInfo object provides details for every session that terminates at the CCAP Core from the CCAP Core's point of view. There will be multiple sessions for each RPD with which the CCAP Core is associated. A row entry is created for every session the CCAP Core terminates.

The CCAP Core MUST populate the CcapL2tpSessionInfo table with each DEPI, UEPI, OOB, NDF, and NDR pseudowire that it terminates.

Table 7-15 - CcapL2tpSessionInfo Object

Attribute Name	Type	Access	Type Constraints	Units
SessionIpAddrType	InetAddressType	Key		N/A
CcapLcceIpAddress	InetAddress	Key		N/A
LocalL2TpSessionId	UnsignedInt	Key		N/A

Table 7-16 - CcapL2tpSessionInfo Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
SessionInfo	Specialization of SessionInfo			
CinLatency	Directed composition to CinLatency	1	1	
SessionCinLatencyPerfTable	Directed composition to SessionCinLatencyPerfTable	1	1	

7.1.11.1 SessionIpAddrType

Indicates whether the IP address provide in the CcapLcceIpAddress and RemoteLcceIpAddr attributes are IPv4 or IPv6.

7.1.11.2 CcapLcceIpAddress

Provides the local LCCE IP address on the CCAP Core of the session detailed in the row entry.

7.1.11.3 LocalL2tpSessionId

Provides the value of the session ID assigned to the session by the CCAP Core.

7.1.12 SessionInfo[30]

This abstract object is based on the docsIfMCmtsDepiSessionInfo object defined in the DOCS-IF-M-CMTS-MIB and the PW3 MIB definition in [RFC 5601]. They have been extended for Remote PHY.

The attributes in this abstract class are used to create an entry for each L2TPv3 tunnel (session) terminated at an R-PHY entity (either RPD or CCAP Core). The sessions terminated in the RPD are provided in the RpdL2tpSessionInfo object; the sessions terminated in the CCAP Core are provided in the CcapL2tpSessionInfo object.

[29] Added by R-OSSI-N-16.1443-2 on 4/22/16 by KB.
[30] Section and sub-sections modified per R-OSSI-N-16.1443-2 on 4/22/16 by KB.

Table 7-17 - SessionInfo Object

Attribute Name	Type	Access	Type Constraints	Units
RemoteLcceIpAddr	InetAddress	read-only		N/A
RemoteL2tpSessionId	unsignedInt	read-only		
CoreId	MacAddress	read-only		N/A
ConnCtrlId	unsignedInt	read-only		N/A
UdpPort	InetPortNumber	read-only		N/A
Description	string	read-only		N/A
SessionType	enum	read-only	psp(1), mpt(2)	N/A
SessionSubType	enum	read-only	mptLegacy (1), pspLegacy(2), mcm(3), pspDepiMultichannel(4), pspUepiScQam(5), pspUepiOfdma(6), pspBwReqScQam(7), pspBwReqOfdma(8), pspProbe(9), pspRngReqScQam(10), pspRngReqOfdma(11), pspMapScQam(12), pspMapOfdma(13), pspSpecman(14), pspPnm(15) psp551Fwd(16), psp551Ret(17), psp552Fwd(18), psp552Ret(19), pspNdf(20), pspNdr(21)	
MaxPayload	unsignedInt	read-only		
PathPayload	unsignedInt	read-only		
RpdIfMtu	unsignedInt	read-only		N/A
CoreIfMtu	unsignedInt	read-only		N/A
IncludeDOCSISMsgs	boolean	read-only		N/A
ErrorCode	enum	read-only	none(1), invalidMACInterfaceValue(2), invalidInterfaceValue(3), noResourcesForInterfaceIndex(4), l2tpv3Error(5), ifAdminStatusSetToDown(6)	
CreationTime	TimeStamp	read-only		N/A
OperStatus	enum	read-only	other(0), up(1), down(2), testing(3), dormant(4), notPresent(5), lowerLayerDown(6)	N/A

Attribute Name	Type	Access	Type Constraints	Units
LocalStatus	enumBits	read-only	other(0), pwNotForwarding(1), servicePwRxFault(2), service PwTxFault(3), psnPwRxFault(4), psnPwTxFault(5)	N/A
LastChange	TimeTicks	read-only		N/A

Table 7-18 - SessionInfo Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
SessionStats	Directed composition to SessionStats	1	1	

7.1.12.1 RemoteLcceIpAddr

Provides the remote-side LCCE IP address of the session detailed in the row entry.

7.1.12.2 RemoteL2tSessionId

This attribute holds the value of the session ID assigned to the session by the remote peer (either RPD or CCAP Core).

7.1.12.3 CoreId

The MAC address of the CCAP Core with which this session terminates. The CCAP-Core sends its MAC address to the RPD periodically via the DOCSIS SYNC MAC Message in the Source Address field.

7.1.12.4 ConnCtrlId

Indicates the control connection identifier (CCID) for this session.

7.1.12.5 UdpPort

The UDP Port reported by the RPD when the DEPI session uses L2TPv3 Header Over UDP. This attribute reports a value of 0 when the session is running with the L2TPv3 Session IP Header.

This port number is negotiated between the CCAP Core and the RPD according to [RFC 3931].

7.1.12.6 Description

Provides an ASCII string constructed with the form:

RemoteEndId=(*pp:mmm*:*ccc*),… { repeated for multiple endpoints }
where
pp is the 0-based port number signaled in the RemoteEndId AVP
mmm is the channel-type enum value from the RemoteEndId AVP
ccc is the channel number from the RemoteEndId AVP

7.1.12.7 SessionType

This attribute specifies whether the session is an MPT session or PSP session.

7.1.12.8 SessionSubType

This specifies the type of DEPI MPT or DEPI PSP session.

7.1.12.9 MaxPayload

The maximum MTU negotiated between the CCAP Core and the RPD during the session establishment process. It considers the header subtractions as indicated in the [R-DEPI] specification.

7.1.12.10 PathPayload

The maximum MTU traversing the CIN from CCAP Core to the RPD. This is calculated by the CCAP Core by procedures such as MTU discovery as described in the [R-PHY] specification.

7.1.12.11 RpdIfMtu

Provides the RPD's CIN interface MTU and is read as the value of the following L2TPV3 AVP transmitted by the RPD during session setup:

Table 7-19 - RpdIfMtu Values

DEPI Downstream PW	UEPI Upstream PW
DEPI Remote MTU AVP (ICRP)	UEPI Remote MTU AVP (ICRP)

7.1.12.12 CoreIfMtu

Provides the CCAP-Core's CIN interface MTU and is read as the value of the following L2TPv3 AVP as received by the device during session setup:

Table 7-20 - CoreIfMtu Values

DEPI Downstream PW	UEPI Upstream PW
DEPI Local MTU AVP (ICRQ)	UEPI Local MTU AVP (ICRQ)

7.1.12.13 IncludeDOCSISMsgs

Reports true if the CCAP Core includes DOCSIS MAP messages and other MAC Management messages in the interface entry associated with this control entry. The CCAP Core determines whether the interface includes DOCSIS messages as part of the payload.

7.1.12.14 ErrorCode

The error Code raised when the session is in error state.

'invalidMACInterfaceValue' indicates wrong assignment of the CCAP Core MAC interface ifIndex.

'invalidInterfaceValue' indicates wrong assignment of the CCAP Core Downstream interface ifIndex.

'noResourcesForInterfaceIfIndex' indicates the CCAP Core has no more resources to assign a session to this entry.

'l2tpv3Error' indicates an L2TPv3 StopCCN or CDN message was issued.

7.1.12.15 CreationTime

The sysUptime when the entry was turned active.

7.1.12.16 OperStatus

Provides the current status of the pseudowire from the point of view of the specific reporting entity (either CCAP Core or RPD).

Values are as follows:

'other' indicates a vendor-specific operational status.

'up' indicates that the pseudowire is ready to pass packets.

'down' indicates that pseudowire signaling is not yet finished, or indications available at the service level indicate that the pseudowire is not passing packets.

'testing' indicates that AdminStatus at the pseudowire level is set to test.

'dormant' indicates that the pseudowire is not in a condition to pass packets but is in a 'pending' state, waiting for some external event.

'notPresent' indicates that some component is missing to accomplish the setup of the pseudowire. It can be a configuration error, incomplete configuration, or a missing H/W component.

'lowerLayerDown' indicates one or more of the lower-layer interfaces responsible for running the underlying PSN is not in OperStatus 'up' state."

7.1.12.17 LocalStatus

Provides the status of the pseudowire in the local node. If the 'other' bit is set, it indicates that an additional vendor-specific status is reported. If none of the bits are set, it indicates no faults are reported.

7.1.12.18 LastChange

Provides the value of sysUpTime when the session entered its current OperStatus state from the point of view of the specific reporting entity (either CCAP Core or RPD).

7.1.13 SessionStats

This table holds performance statistics for the referenced session. The SessionStats object is based on the docsIfMCmtsDepiSessionStats object defined in the DOCS-IF-M-CMTS-MIB and has been extended for Remote PHY.

The CCAP Core MUST populate the SessionStats table with each DEPI, UEPI, OOB, NDF, and NDR pseudowire that is established on each RPD with which it is connected.

If the RDP supports SNMP, the RPD SHOULD populate the SessionStats table with each DEPI, UEPI, OOB, NDF, and NDR pseudowire it has established with CCAP Cores.

Table 7-21 - SessionStats Object

Attribute Name	Type	Access	Type Constraints	Units
OutOfSequencePackets	counter32	Read-only		Packets

7.1.13.1 OutOfSequencePackets

The count of session packets that were received out of sequence from the point of view of the reporting entity.

It is vendor dependent the re-sequence of packets. Implementations that do not re-sequence packets also increase the value of ifInDiscards for the respective entry.

7.1.14 CinLatency

These objects provide a measurement on the latency on the CIN link for that session as measured by the CCAP Core. These measurements are based on the DEPI latency measurement, specified in [R-DEPI]. The CCAP Core SHOULD implement the CinLatency object.

Table 7-22 - CinLatency Object

Attribute Name	Type	Access	Type Constraints	Units
CinLastValue	unsignedInt	Read-only		Master clock ticks
CinLatencyValueLastTime	TimeTicks	Read-only		
CinLatencyInterval	unsignedInt	Read-only		seconds

7.1.14.1 CinLastValue

The latest latency measurement on this session.

7.1.14.2 CinLatencyValueLastTime

The sysUpTime value of the last time the CinLastValue object was updated.

7.1.14.3 CinLatencyInterval

The time interval used to measure periodically the CIN latency per DEPI session. Active measurement of CIN latency applies to active DEPI sessions only.

This object is constrained to 420 seconds to prevent Master Clock counter overruns. A value zero indicates no CIN latency measurements are configured to be performed.

7.1.15 SessionCinLatencyPerfTable

This table provides cumulative measurements of the CIN latency on the network as measured by the CCAP Core. When the table is full, the oldest measurement is replaced with a new one. The SessionCinLatencyPerfTable object is based on the docsIfMCmtsDepiSessionCinLatencyPerfTable object defined in the DOCS-IF-M-CMTS-MIB and has been extended for Remote PHY.

The CCAP Core SHOULD implement the SessionCinLatencyPerfTable object.

Table 7-23 - SessionCinLatencyPerfTable Object

Attribute Name	Type	Access	Type Constraints	Units
IntervalSeq	unsignedInt	Read-only		
Value	unsignedInt	Read-only		
MeasTime	TimeTicks	Read-only		

7.1.15.1 IntervalSeq

The interval sequence where the CIN latency measurement was taken. It is valid in an implementation that overrides the oldest sequence number entry with the most recent measurement.

7.1.15.2 Value

The CIN latency value measured for the session referenced by this entry.

7.1.15.3 MeasTime

The sysUpTime value the last time this latency measurement was updated.

7.1.16 EnetIfTable

This object provides details about the Ethernet interfaces on the RPD. The objects in this table are based on the ifTable/ifXTable specified in [RFC 2863].The CCAP Core MUST implement a row entry in this table for every Ethernet interface on the RPD.

Table 7-24 - EnetIfTable Object

Attribute Name	Type	Access	Type Constraints	Units
ifIndex	ifIndex	key		N/A
ifName	unsignedByte	Read-Only		N/A
ifDescr	string	Read-Only		N/A
ifType	IANAifType	Read-Only		N/A
ifAlias	string	Read-Only		N/A

Attribute Name	Type	Access	Type Constraints	Units
ifMTU	integer	Read-Only		N/A
ifPhysAddress	PhysAddres	Read-Only		N/A
ifAdminStatus	enum	Read-Only	up(1), down(2), testing(3)	N/A
ifOperStatus	enum	Read-Only	up(1), down(2), testing(3), unknown(4), dormant(5), notPresent(6), lowerLayerDown(7)	N/A
ifLastChange	TimeTicks	Read-Only		N/A
IfInOctets	counter64	Read-Only		octets
ifInUnicastOctets	counter64	Read-Only		octets
inInMulticastOctets	counter64	Read-Only		octets
ifInFrames	counter64	Read-Only		frames
ifInUnicastFrames	counter64	Read-Only		frames
ifInMulticastFrames	counter64	Read-Only		frames
ifInBroadcastFrames	counter64	Read-Only		frames
ifOutOctets	counter64	Read-Only		octets
ifOutUnicastOctets	counter64	Read-Only		octets
ifOutMulticastOctets	counter64	Read-Only		octets
ifOutFrames	counter64	Read-Only		frames
ifOutUnicastFrames	counter64	Read-Only		frames
ifOutMulticastFrames	counter64	Read-Only		frames
ifOutBroadcastFrames	counter64	Read-Only		frames
ifHighSpeed	guage32	Read-Only		mbps
ifLinkUpDownTrapEnable	integer	Read-Only		N/A
ifPromiscuousMode	boolean	Read-Only		N/A
ifConnectorPresent	boolean	Read-Only		N/A
ifCounterDiscontinuity	TimeStamp	Read-Only		N/A

7.1.16.1 IfIndex

Unique index for this Ethernet interface.

7.1.16.2 ifName

A name that describes the interface. The CCAP Core MUST populate this object with the ID that is used in GCP for this port.

7.1.16.3 ifDescr

A textual string containing information about the Ethernet interface. This string should include the name of the manufacturer, the product name and the version of the interface hardware/software.

7.1.16.4 ifType

The type of interface. Additional values for ifType are assigned by the Internet Assigned Numbers Authority (IANA), through updating the syntax of the IANAifType textual convention. The types are defined in the IANAifType-MIB.

7.1.16.5 ifAlias

On the first instantiation of an interface, the value of ifAlias associated with that interface is the zero-length string. As and when a value is written into an instance of ifAlias through a network management operation, then the agent retains the supplied value in the ifAlias instance associated with the same interface for as long as that interface remains instantiated, including across all re-initializations/reboots of the network management system, including those which result in a change of the interface's ifIndex value.

7.1.16.6 ifMTU

The size of the largest packet which can be sent/received on the interface, specified in octets. For interfaces that are used for transmitting network datagrams, this is the size of the largest network datagram that can be sent on the interface.

7.1.16.7 ifPhysAddress

The interface's address at its protocol sub-layer. For example, for an 802.x interface, this object normally contains a MAC address. The interface's media-specific MIB defines the bit and byte ordering and the format of the value of this object. For interfaces which do not have such an address (e.g., a serial line), this object should contain an octet string of zero length.

7.1.16.8 ifAdminStatus

The state of the interface. The testing(3) state indicates that no operational packets can be passed. When a managed system initializes, all interfaces start with ifAdminStatus in the down(2) state. As a result of either explicit management action or per configuration information retained by the managed system, ifAdminStatus is then changed to either the up(1) or testing(3) states (or remains in the down(2) state).

7.1.16.9 ifOperStatus

The current operational state of the interface. The testing(3) state indicates that no operational packets can be passed. If ifAdminStatus is down(2) then ifOperStatus should be down(2). If ifAdminStatus is changed to up(1), then ifOperStatus should change to up(1) if the interface is ready to transmit and receive network traffic; it should change to dormant(5) if the interface is waiting for external actions (such as a serial line waiting for an incoming connection); it should remain in the down(2) state if and only if there is a fault that prevents it from going to the up(1) state; it should remain in the notPresent(6) state if the interface has missing (typically, hardware) components.

7.1.16.10 ifLastChange

The value of sysUpTime at the time the interface entered its current operational state. If the current state was entered prior to the last re-initialization of the local network management subsystem, then this object contains a zero value.

7.1.16.11 ifInOctets

This attribute is the count of all octets received by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.12 IfInUnicastOctets

This attribute is the count of all unicast octets received by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.13 ifInMulticastOctets

This attribute is the count of all multicast octets received by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.14 ifInFrames

This attribute is the count of all frames received by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.15 ifInUnicastFrames

This attribute is the count of all unicast frames received by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.16 ifInMulticastFrames

This attribute is the count of all multicast frames received by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.17 ifInBroadcastFrames

This attribute is the count of all broadcast frames received by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.18 ifOutOctets

This attribute is the count of all octets transmitted by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.19 ifOutUnicastOctets

This attribute is the count of all unicast octets transmitted by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.20 ifOutMulticastOctets

This attribute is the count of all multicast octets transmitted by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.21 ifOutFrames

This attribute is the count of all frames transmitted by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.22 ifOutUnicastFrames

This attribute is the count of all unicast frames transmitted by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.23 ifOutMulticastFrames

This attribute is the count of all multicast frames transmitted by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.24 ifOutBroadcastFrames

This attribute is the count of all broadcast frames transmitted by the RPD on this Ethernet interface. Discontinuities in the value of this counter can occur at reinitialization of the managed system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

7.1.16.25 ifHighSpeed

An estimate of the interface's current bandwidth in units of 1,000,000 bits per second. If this object reports a value of `n' then the speed of the interface is somewhere in the range of `n-500,000' to `n+499,999'. For interfaces which do not vary in bandwidth or for those where no accurate estimation can be made, this object should contain the nominal bandwidth.

7.1.16.26 ifLinkUpDownTrapEnable

Indicates whether linkup/linkdown traps are generated for this interface.

A value of '1' indicates that traps are enabled.

A value of '2' indicates that traps are disabled.

7.1.16.27 ifPromiscuousMode

This object has a value of '2' (false) if this interface only accepts packets/frames that are addressed to this interface.

This object has a value of '1' (true) when the station accepts all packets/frames transmitted on the media.

The value of ifPromiscuousMode does not affect the reception of broadcast and multicast packets/frames by the interface.

7.1.16.28 ifConnectorPresent

This object has the value 'true' if the interface sublayer has a physical connector and the value 'false' otherwise.

7.1.16.29 ifCounterDiscontinuity

The value of sysUpTime on the most recent occasion at which any one or more of this interface's counters suffered a discontinuity. The relevant counters are the specific instances associated with this interface of any Counter64 object contained in this table. If no such discontinuities have occurred since the last re-initialization of the local management subsystem, then this object contains a zero value.

7.1.17 IP MIB Objects

A subset of MIB objects from the IP-MIB [RFC 4293] are required for reporting on the IP interfaces on the RPD. The objects in the following sub-sections are from [RFC 4293] and used here with modifications.

7.1.17.1 IpInterface

This table is a collection of objects that describe IP forwarding versions supported and provide status details.

Table 7-25 - IpInterface Object

Attribute Name	Type	Access	Type Constraints	Units
Ipv4InterfaceTableLastChange	TimeStamp	Read-only		N/A
Ipv6InterfaceTableLastChange	TimeStamp	Read-only		N/A
IpIfStatsTableLastChange	TimeStamp	Read-only		N/A

Attribute Name	Type	Access	Type Constraints	Units
Ipv4DefaultTTL	integer	Read-only	1..255	N/A
Ipv4ReasmTimeout	integer	Read-only		seconds
Ipv6DefaultHopLimit	integer	Read-only	0..255	N/A

Table 7-26 - IpInterface Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
Ipv4Interfaces	Directed composition to Ipv4Interfaces	1	0..*	
Ipv6Interfaces	Directed composition to Ipv6Interfaces	1	0..*	
IpIfStats	Directed composition to IpIfStats	1	0..*	
IpAddresses	Directed composition to IpAddresses	1	0..*	
IpNetToPhysical	Directed composition to IpNetToPhysical	1	0..*	
IpDefaultRouter	Directed composition to IpDefaultRouter	1	0..*	
Ipv6RouterAdvertisement	Directed composition to Ipv6RouterAdvertisement	1	0..*	
IcmpStats	Directed composition to IcmpStats	1	0..2	
IcmpMsgStats	Directed composition to IcmpMsgStats	1	0..*	

7.1.17.1.1 Ipv4InterfaceTableLastChange

The value of sysUpTime on the most recent occasion at which a row in the Ipv4Interfaces table was added or deleted, or when a ReasmMaxSize or an EnableStatus object in the Ipv4Interfaces table was modified.

7.1.17.1.2 Ipv6InterfaceTableLastChange

The value of sysUpTime on the most recent occasion at which a row in the Ipv6Interfaces table was added or deleted or when a ReasmMaxSize, InterfaceIdentifier, EnableStatus, ReachableTime, RetransmitTime, or Forwarding object in the Ipv6Interfaces table was modified.

7.1.17.1.3 IpIfStatsTableLastChange

The value of sysUpTime on the most recent occasion at which a row in the ipIfStatsTable was added or deleted.

7.1.17.1.4 Ipv4DefaultTTL

The default value inserted into the Time-To-Live field of the IPv4 header of datagrams originated at this entity, whenever a TTL value is not supplied by the transport layer protocol.

7.1.17.1.5 Ipv4ReasmTimeout

The maximum number of seconds that received fragments are held while they are awaiting reassembly at this entity.

7.1.17.1.6 Ipv6IpDefaultHopLimit

The default value inserted into the Hop Limit field of the IPv6 header of datagrams originated at this entity whenever a Hop Limit value is not supplied by the transport layer protocol.

7.1.17.2 Ipv4Interfaces

This table provides details on the IPv4 interfaces on the RPD. It is modeled after the ipv4InterfaceTable specified in [RFC 4293].

Table 7-27 - Ipv4Interfaces Object

Attribute Name	Type	Access	Type Constraints	Units
ifIndex	ifIndex	key		N/A
ReasmMaxSize	integer	Read-only		N/A
EnableStatus	integer	Read-only	1..2	N/A
RetransmitTime	integer	Read-only		milliseconds

7.1.17.2.1 ifIndex

The index value that uniquely identifies the IPv4 interface to which this entry is applicable.

7.1.17.2.2 ReasmMaxSize

The size of the largest IPv4 datagram that this entity can re-assemble from incoming IPv4 fragmented datagrams received on this interface.

7.1.17.2.3 EnableStatus

The indication of whether IPv4 is enabled (up) or disabled (down) on this interface.

'1' indicates that IPv4 is enabled.

'2' indicates that IPv4 is disabled.

7.1.17.2.4 RetransmitTime

The time between retransmissions of ARP requests to a neighbor when resolving the address or when probing the reachability of a neighbor.

7.1.17.3 Ipv6Interfaces

This table provides details on the IPv6 interfaces on the RPD. It is modeled after the ipv6InterfaceTable specified in [RFC 4293].

Table 7-28 - Ipv6Interfaces Object

Attribute Name	Type	Access	Type Constraints	Units
ifIndex	ifIndex	key		N/A
ReasmMaxSize	integer	Read-only		N/A
InterfaceIdentifier	octetString	Read-only		N/A
EnableStatus	integer	Read-only	1..2	N/A
ReachableTime	unsignedInt	Read-only		milliseconds
RetransmitTime	unsignedint	Read-only		milliseconds

7.1.17.3.1 ifIndex

The index value that uniquely identifies the Ipv6 interface to which this entry is applicable.

7.1.17.3.2 ReasmMaxSize

The size of the largest IPv6 datagram that this entity can re-assemble from incoming IPv6 fragmented datagrams received on this interface.

7.1.17.3.3 InterfaceIdentifier

The Interface Identifier for this interface. The Interface Identifier is combined with an address prefix to form an interface address.

By default, the Interface Identifier is auto-configured according to the rules of the link type to which this interface is attached.

A zero length identifier may be used where appropriate. One possible example is a loopback interface.

7.1.17.3.4 EnableStatus

The indication of whether IPv6 is enabled (up) or disabled (down) on this interface.

'1' indicates that IPv6 is enabled.

'2' indicates that IPv6 is disabled.

7.1.17.3.5 ReachableTime

The time a neighbor is considered reachable after receiving a reachability confirmation.

7.1.17.3.6 RetransmitTime

The time between retransmissions of Neighbor Solicitation messages to a neighbor when resolving the address or when probing the reachability of a neighbor.

7.1.17.4 IpIfStats

This table contains per-interface traffic statistics. It is modeled after the IP-MIB IpIfStatsTable described in [RFC 4293]; however, all counters are 64 bit.

Table 7-29 - IpIfStats Object

Attribute Name	Type	Access	Type Constraints	Units
IpVersion	InetVersion	key		N/A
ifIndex	ifIndex	key		N/A
InReceives	counter64	Read-only		datagrams
InOctets	counter64	Read-only		octets
InHdrErrors	counter64	Read-only		datagrams
InAddrErrors	counter64	Read-only		datagrams
InUnknownProtos	counter64	Read-only		datagrams
InTruncatedPkts	counter64	Read-only		datagrams
InDiscards	counter64	Read-only		datagrams
InDelivers	counter64	Read-only		datagrams
OutRequests	counter64	Read-only		datagrams
OutDiscards	counter64	Read-only		datagrams
OutTransmits	counter64	Read-only		datagrams
OutOctets	counter64	Read-only		octets
InMcastPkts	counter64	Read-only		datagrams
InMcastOctets	counter64	Read-only		octets
OutMcastPkts	counter64	Read-only		datagrams
OutMcastOctets	counter64	Read-only		octets
InBcastPkts	counter64	Read-only		datagrams
OutBcastPkts	counter64	Read-only		datagrams
DiscontinuityTime	TimeStamp	Read-only		N/A

Attribute Name	Type	Access	Type Constraints	Units
RefreshRate	unsignedInt	Read-only		milliseconds

7.1.17.4.1 IpVersion

The IP version of this table row.

7.1.17.4.2 ifIndex

The index value that uniquely identifies the interface to which this entry is applicable.

7.1.17.4.3 InReceives

The total number of input IP datagrams received, including those received in error.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.4 InOctets

The total number of octets received in input IP datagrams, including those received in error.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.5 InHdrErrors

The number of input IP datagrams discarded due to errors in their IP headers, including version number mismatch, other format errors, hop count exceeded, errors discovered in processing their IP options, etc.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.6 InAddrErrors

The number of input IP datagrams discarded because the IP address in their IP header's destination field was not a valid address to be received at this entity. This count includes invalid addresses (e.g., ::0). For entities that are not IP routers and therefore do not forward datagrams, this counter includes datagrams discarded because the destination address was not a local address.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.7 InUnknownProtos

The number of locally-addressed IP datagrams received successfully but discarded because of an unknown or unsupported protocol.

When tracking interface statistics, the counter of the interface to which these datagrams were addressed is incremented. This interface might not be the same as the input interface for some of the datagrams.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.8 InTruncatedPkts

The number of input IP datagrams discarded because the datagram frame didn't carry enough data.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.9 InDiscards

The number of input IP datagrams for which no problems were encountered to prevent their continued processing, but were discarded (e.g., for lack of buffer space). The RPD discards all packets requiring reassembly and those packets are also counted here.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.10 InDelivers

The total number of datagrams successfully delivered to IPuser-protocols (including ICMP).

When tracking interface statistics, the counter of the interface to which these datagrams were addressed is incremented. This interface might not be the same as the input interface for some of the datagrams.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.11 OutRequests

The total number of IP datagrams that local IP user-protocols (including ICMP) supplied to IP in requests for transmission.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.12 OutDiscards

The number of output IP datagrams for which no problem was encountered to prevent their transmission to their destination, but were discarded (e.g., for lack of buffer space). Note that this counter would include datagrams that required fragmentation.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.13 OutTransmits

The total number of IP datagrams that this entity supplied to the lower layers for transmission.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.14 OutOctets

The total number of octets in IP datagrams delivered to the lower layers for transmission. Octets from datagrams counted in the OutTransmits object are required to be counted here.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.15 InMcastPkts

The number of IP multicast datagrams received.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.16 InMcastOctets

The total number of octets received in IP multicast datagrams. Octets from datagrams counted in the McastPkts object are required to be counted here.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.17 OutMcastPkts

The number of IP multicast datagrams transmitted.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of the DiscontinuityTime object in this table.

7.1.17.4.18 OutMcastOctets

The total number of octets transmitted in IP multicast datagrams. Octets from datagrams counted in the OutMcastPkts object are required to be counted here.

7.1.17.4.19 InBcastPkts

The number of IP broadcast datagrams received.

The total number of octets transmitted in IP multicast datagrams. Octets from datagrams counted in the OutMcastPkts object are required to be counted here.

7.1.17.4.20 OutBcastPkts

The number of IP broadcast datagrams transmitted.

The total number of octets transmitted in IP multicast datagrams. Octets from datagrams counted in the OutMcastPkts object are required to be counted here.

7.1.17.4.21 DiscontinuityTime

The value of sysUpTime on the most recent occasion at which any one or more of this entry's counters suffered a discontinuity.

If no such discontinuities have occurred since the last re-initialization of the local management subsystem, then this object contains a zero value.

7.1.17.4.22 RefreshRate

The minimum reasonable polling interval for this entry. This object provides an indication of the minimum amount of time required to update the counters in this entry.

7.1.17.5 IpAddresses

This table contains addressing information relevant to the RPD's interfaces.

This table does not contain multicast address information.

Note well: When including IPv6 link-local addresses in this table, the entry uses an InetAddressType of 'ipv6z' in order to differentiate between the possible interfaces.

This table is based on the ipAddressTable object specified in [RFC 4293].

Table 7-30 - IpAddresses Object

Attribute Name	Type	Access	Type Constraints	Units
AddrType	InetAddressType	key		N/A
IpAddress	InetAddress	key		N/A
ifIndex	ifIndex	Read-Only		N/A
Type	integer	Read-Only		N/A
Prefix	RowPointer	Read-Only		N/A

Attribute Name	Type	Access	Type Constraints	Units
Origin	enum	Read-Only	other(1), manual(2), wellKnown(3), dhcp(4), routerAdv(5)	N/A
Status	enum	Read-Only	preferred(1), deprecated(2), invalid(3), inaccessible(4), unknown(5), tentative(6), duplicate(7), optimistic(8)	N/A
Created	TimeStamp	Read-Only		N/A
LastChanged	TimeStamp	Read-Only		N/A

Table 7-31 - IpAddresses Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
EnetIfTable	Directed association to EnetIfTable	0..*		ifIndex

7.1.17.5.1 AddrType

The IP address type of the IpAddress object.

7.1.17.5.2 IpAddress

The IP address to which this entry's addressing information pertains. The address type of this object is specified in the AddrType object.

7.1.17.5.3 ifIndex

The index value that uniquely identifies the interface on which this IP address appears. When the IP appears on an Ethernet interface, this is the same value as the ifIndex in the EnetIfTable.

7.1.17.5.4 Type

The type of traffic for which the address can be used.

The value '1' indicates the address type is unicast.

The value '2' indicates the address type is anycast.

The value '3' indication the address type is broadcast. This is not a valid value for IPv6 addresses.

7.1.17.5.5 Origin

The origin of this IP address.

'manual' indicates an IP address that was manually configured.

'wellKnown' indicates a well-known IP address. Well known prefixes may be assigned by IANA, the address registries, or by specification in a standards track RFC.

'dhcp' indicates an IP address that was assigned by a DHCP server.

'routerAdv' indicates an IP address learned from a router.

'other' indicates an origin not covered by the options here.

7.1.17.5.6 Status

The status of an address. Most of the states correspond to states from the IPv6 Stateless Address Autoconfiguration protocol.

The value 'preferred' indicates that this is a valid address that can appear as the destination or source address of a packet.

The value 'deprecated' indicates that this is a valid but deprecated address that should no longer be used as a source address in new communications, but packets addressed to such an address are processed as expected.

The value 'invalid' indicates that this isn't a valid address and it shouldn't appear as the destination or source address of a packet.

The value 'inaccessible' indicates that the address is not accessible because the interface to which this address is assigned is not operational.

The value 'unknown' indicates that the status cannot be determined for some reason.

The value 'tentative' indicates that the uniqueness of the address on the link is being verified. Addresses in this state should not be used for general communication and should only be used to determine the uniqueness of the address.

The value 'duplicate' indicates the address has been determined to be non-unique on the link and so cannot be used.

The value 'optimistic' indicates the address is available for use, subject to restrictions, while its uniqueness on a link is being verified.

In the absence of other information, an IPv4 address is always 'preferred'.

7.1.17.5.7 Created

The value of sysUpTime at the time this entry was created. If this entry was created prior to the last re-initialization of the local network management subsystem, then this object contains a zero value.

7.1.17.5.8 LastChanged

The value of sysUpTime at the time this entry was last updated. If this entry was updated prior to the last re-initialization of the local network management subsystem, then this object contains a zero value.

7.1.17.6 IpNetToPhysical

The IP Address Translation table used for mapping from IP addresses to physical addresses.

The Address Translation tables contain the IP address to 'physical' address equivalences.

While many protocols may be used to populate this table, ARP and Neighbor Discovery are the most likely options.

This table is based on the ipNetToPhysicalTable object specified in [RFC 4293].

Table 7-32 - IpNetToPhysical Object

Attribute Name	Type	Access	Type Constraints	Units
ifIndex	ifIndex	key		N/A
AddrType	InetAddressType	key		N/A
NetAddress	InetAddress	key		N/A
PhysAddress	PhysAddress	Read-Only		N/A
LastUpdated	TimeStamp	Read-Only		N/A

Attribute Name	Type	Access	Type Constraints	Units
Type	enum	Read-Only	other(1), invalid(2), dynamic(3), static(4), local(5)	N/A
State	enum	Read-Only	reachable(1), stale(2), delay(3), probe(4), invalid(5), unknown(6) incomplete(7)	N/A

7.1.17.6.1 ifIndex

The index value that uniquely identifies the interface to which this entry is applicable.

7.1.17.6.2 AddrType

The type of IP address in the NetAddress object.

7.1.17.6.3 NetAddress

The IP address corresponding to the media-dependent 'physical' address. The address type of this object is specified in the AddrType object.

7.1.17.6.4 PhysAddress

The media-dependent 'physical' address.

7.1.17.6.5 LastUpdated

The value of sysUpTime at the time this entry was last updated. If this entry was updated prior to the last re-initialization of the local network management subsystem, then this object contains a zero value.

7.1.17.6.6 Type

Specifies the type of mapping.

It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations have to be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant Type object.

The value 'invalid' indicates that this is an invalidated mapping.

The value 'dynamic' indicates that the IP address to physical addresses mapping has been dynamically resolved using a protocol such as IPv4 ARP or the IPv6 Neighbor Discovery protocol.

The value 'static' indicates that the mapping has been statically configured. Both of these refer to entries that provide mappings for other entities addresses.

The value 'local' indicates that the mapping is provided for an entity's own interface address.

The value 'other' indicates that none of these defined types applies to this mapping.

7.1.17.6.7 State

The Neighbor Unreachability Detection state for the interface when the address mapping in this entry is used. If Neighbor Unreachability Detection is not in use (e.g., for IPv4), this object is always unknown(6).

The value 'reachable' indicates confirmed reachability.

The value 'stale' indicates unconfirmed reachability.

The value 'delay' indicates that the protocol is waiting for reachability confirmation before entering the probe state.

The value 'probe' indicates active probing.

The value 'invalid' indicates an invalidated mapping.

The value 'unknown' indicates the state cannot be determined for some reason.

The value 'incomplete' indicates that address resolution is being performed.

7.1.17.7 IpDefaultRouter

This table used to describe the default routers known to the RPD. It is based on the ipDefaultRouterTable object specified in [RFC 4293].

Table 7-33 - IpDefaultRouter Object

Attribute Name	Type	Access	Type Constraints	Units
AddrType	InetAddressType	key		N/A
RouterIpAddress	InetAddress	key		N/A
ifIndex	ifIndex	key		N/A
Lifetime	unsignedInt	Read-Only		seconds
Preference	integer	Read-Only		N/A

7.1.17.7.1 AddrType

The IP address type for this row of the table.

7.1.17.7.2 RouterIpAddress

The IP address of the default router represented by this row. The address type of this object is specified in the AddrType object.

7.1.17.7.3 ifIndex

The index value that uniquely identifies the interface by which the router can be reached.

7.1.17.7.4 Lifetime

The remaining length of time, in seconds, that this router will continue to be useful as a default router. A value of zero indicates that it is no longer useful as a default router. It is left to the implementer of the MIB as to whether a router with a lifetime of zero is removed from the list.

For IPv6, this value should be extracted from the router advertisement messages.

7.1.17.7.5 Preference

An indication of preference given to this router as a default router as described in the Default Router Preferences document. Treating the value as a 2-bit signed integer allows for simple arithmetic comparisons.

For IPv4 routers or IPv6 routers that are not using the updated router advertisement format, this object is set to medium (0).

The value '-2' is reserved.

The value '-1' indicates low preference.

The value '0' indicates medium preference.

The value '1' indicates high preference.

7.1.17.8 IcmpStats

This table provides generic system-wide ICMP counters. It is based on the icmpStatsTable object specified in [RFC 4293].

Table 7-34 - IcmpStats Object

Attribute Name	Type	Access	Type Constraints	Units
IpVersion	InetVersion	key		N/A
InMsgs	counter64	Read-Only		N/A
InErrors	counter64	Read-Only		N/A
OutMsgs	counter64	Read-Only		N/A
OutErrors	counter64	Read-Only		N/A

7.1.17.8.1 IpVersion

The IP version of the statistics. Statistics are provided for each IP version supported.

7.1.17.8.2 InMsgs

The total number of ICMP messages that the entity received. Note that this counter includes all those counted by the InErrors object.

7.1.17.8.3 InErrors

The number of ICMP messages that the entity received but determined as having ICMP-specific errors (bad ICMP checksums, bad length, etc.).

7.1.17.8.4 OutMsgs

The total number of ICMP messages that the entity attempted to send. Note that this counter includes all those counted by the OutErrors object.

7.1.17.8.5 OutErrors

The number of ICMP messages that this entity did not send due to problems discovered within ICMP, such as a lack of buffers. This value should not include errors discovered outside the ICMP layer, such as the inability of IP to route the resultant datagram. In some implementations, there may be no types of error that contribute to this counter's value.

7.1.17.9 IcmpMsgStats

This table provides system-wide per-version, per-message type ICMP counters. It is based on the icmpMsgStatsTable object specified in [RFC 4293].

The system should track each ICMP type value, even if that ICMP type is not supported by the system. However, a given row need not be instantiated unless a message of that type has been processed, i.e., the row for Type=X may be instantiated before but is required to be instantiated after the first message with Type=X is received or transmitted. After receiving or transmitting any succeeding messages with Type=X, the relevant counter is incremented.

Table 7-35 - IcmpMsgStats Object

Attribute Name	Type	Access	Type Constraints	Units
IpVersion	InetVersion	key		N/A
Type	Integer	key		N/A

Attribute Name	Type	Access	Type Constraints	Units
InPkts	counter64	Read-Only		N/A
OutPkts	counter64	Read-Only		N/A

7.1.17.9.1 IpVersion

The IP version of the statistics. Statistics are provided for each IP version supported.

7.1.17.9.2 Type

The ICMP type field of the message type being counted by this row.

Note that ICMP message types are scoped by the address type in use.

7.1.17.9.3 InPkts

The number of input packets for this AF and type.

7.1.17.9.4 OutPkts

The number of output packets for this AF and type.

7.1.18 EntityTable

The EntityTable is implemented in the DOCS-RPHY-MIB to represent entities that exist within the Remote PHY Node that are known to the RPD. Because the entities exist in the Remote PHY Node and not the CCAP Core, the CCAP Core has no knowledge of their existence and have to be reported by the RPD through this MIB.

The CCAP Core MUST implement a row entry in the EntityTable object for each known RPD interface and entity.

The CCAP Core MUST provide the unique component serial number, via the SerialNum object, contained within the row entry in the EntityTable for the enclosure, RPD module, and each FRU that has a serial number in the system. Example FRUs with serial numbers include, but are not limited to, Ethernet cards, RF amplifiers, and RPD modules.

The CCAP Core MUST implement a row entry in the EntityTable for the enclosure with a Class value of "chassis".

The CCAP Core MUST implement a row entry in the EntityTable for the RPD module with a Class value of "module".

The CCAP Core MUST implement row entries in the EntityTable for sensors in the system with a Class value of "sensor".

When a row is added for a sensor in the EntityTable, the CCAP Core SHOULD create a row in the SensorDetails table.

This object is based on the entPhysicalTable object specified in the ENTITY-MIB [RFC 4133].

Table 7-36 - EntityTable Object

Attribute Name	Type	Access	Type Constraints	Units
Index	unsignedInt	key		N/A
Descr	string	Read-Only		N/A
VendorType	AutonomousType	Read-Only		N/A
ContainedIn	PhysicalIndexor Zero	Read-Only		N/A

Attribute Name	Type	Access	Type Constraints	Units
Class	Enum	Read-Only	other(1), unknown(2), chassis(3), backplane(4), container(5), powerSupply(6), fan(7), sensor(8), module(9), port(10), stack(11), cpu(12)	N/A
ParentRelPos	integer	Read-Only		N/A
Name	string	Read-Only		N/A
HardwareRev	string	Read-Only		N/A
FirmwareRev	string	Read-Only		N/A
SoftwareRev	string	Read-Only		N/A
SerialNum	string	Read-Only		N/A
MfgName	string	Read-Only		N/A
ModelName	string	Read-Only		N/A
Alias	string	Read-Only		N/A
AssetId	string	Read-Only		N/A
IsFRU	boolean	Read-Only		N/A
MfgDate	TimeDate	Read-Only		N/A
Uris	OctetString			

Table 7-37 - EntityTable Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
SensorDetails	Directed composition to SensorDetails	1	1	

7.1.18.1 Index

An arbitrary value that uniquely identifies the physical entity. Index values for different physical entities are not necessarily contiguous.

7.1.18.2 Descr

A textual description of physical entity. This object should contain a string that identifies the manufacturer's name for the physical entity, and should be set to a distinct value for each version or model of the physical entity.

7.1.18.3 VendorType

An indication of the vendor-specific hardware type of the physical entity. Note that this is different from the definition of MIB-II's sysObjectID.

An agent should set this object to an enterprise-specific registration identifier value indicating the specific equipment type in detail. The associated instance of Class is used to indicate the general type of hardware device.

If no vendor-specific registration identifier exists for this physical entity, or the value is unknown by this agent, then the value { 0 0 } is returned.

7.1.18.4 ContainedIn

The value of the Index for the physical entity that 'contains' this physical entity. A value of zero indicates this physical entity is not contained in any other physical entity. Note that the set of 'containment' relationships define a strict hierarchy; that is, recursion is not allowed.

In the event that a physical entity is contained by more than one physical entity (e.g., double-wide modules), this object should identify the containing entity with the lowest value of Index.

7.1.18.5 Class

An indication of the general hardware type of the physical entity.

An agent should set this object to the standard enumeration value that most accurately indicates the general class of the physical entity, or the primary class if there is more than one entity.

If no appropriate standard registration identifier exists for this physical entity, then the value 'other(1)' is returned. If the value is unknown by this agent, then the value 'unknown(2)' is returned.

7.1.18.6 ParentRelPos

An indication of the relative position of this 'child' component among all its 'sibling' components. Sibling components are defined as entries that share the same instance values of each of the ContainedIn and Class objects.

An NMS can use this object to identify the relative ordering for all sibling components of a particular parent (identified by the ContainedIn instance in each sibling entry).

If possible, this value should match any external labeling of the physical component. For example, for a container (e.g., card slot) labeled as 'slot #3', ParentRelPos should have the value '3'. Note that the entry for the module plugged in slot 3 should have an ParentRelPos value of '1'.

If the physical position of this component does not match any external numbering or clearly visible ordering, then user documentation or other external reference material should be used to determine the parent-relative position. If this is not possible, then the agent should assign a consistent (but possibly arbitrary) ordering to a given set of 'sibling' components, perhaps based on internal representation of the components.

If the agent cannot determine the parent-relative position for some reason, or if the associated value of ContainedIn is '0', then the value '-1' is returned. Otherwise, a non-negative integer is returned, indicating the parent-relative position of this physical entity.

Parent-relative ordering normally starts from '1' and continues to 'N', where 'N' represents the highest positioned child entity. However, if the physical entities (e.g., slots) are labeled from a starting position of zero, then the first sibling should be associated with a ParentRelPos value of '0'. Note that this ordering may be sparse or dense, depending on agent implementation.

The actual values returned are not globally meaningful, as each 'parent' component may use different numbering algorithms. The ordering is only meaningful among siblings of the same parent component.

The agent should retain parent-relative position values across reboots, either through algorithmic assignment or use of non-volatile storage.

7.1.18.7 Name

The textual name of the physical entity. The value of this object should be the name of the component as assigned by the local device and should be suitable for use in commands entered at the device's `console'. This might be a text name (e.g., `console') or a simple component number (e.g., port or module number, such as `1'), depending on the physical component naming syntax of the device.

If there is no local name, or if this object is otherwise not applicable, then this object contains a zero-length string.

Note that the value of entPhysicalName for two physical entities will be the same in the event that the console interface does not distinguish between them, e.g., slot-1 and the card in slot-1.

7.1.18.8 HardwareRev

The vendor-specific hardware revision string for the physical entity. The preferred value is the hardware revision identifier actually printed on the component itself (if present).

Note that if revision information is stored internally in a non-printable (e.g., binary) format, then the agent converts such information to a printable format, in an implementation-specific manner.

If no specific hardware revision string is associated with the physical component, or if this information is unknown to the agent, then this object will contain a zero-length string.

7.1.18.9 FirmwareRev

The vendor-specific firmware revision string for the physical entity.

Note that if revision information is stored internally in a non-printable (e.g., binary) format, then the agent converts such information to a printable format, in an implementation-specific manner.

If no specific firmware programs are associated with the physical component, or if this information is unknown to the agent, then this object will contain a zero-length string.

7.1.18.10 SoftwareRev

The vendor-specific software revision string for the physical entity.

Note that if revision information is stored internally in a non-printable (e.g., binary) format, then the agent is required to convert such information to a printable format, in an implementation-specific manner.

If no specific software programs are associated with the physical component, or if this information is unknown to the agent, then this object will contain a zero-length string.

7.1.18.11 SerialNum

The vendor-specific serial number string for the physical entity. The preferred value is the serial number string actually printed on the component itself (if present).

Not every physical component will have a serial number, or even need one. Physical entities for which the associated value of the IsFRU object is equal to 'false(2)', do not need their own unique serial number. An agent may return a zero-length string.

Example FRUs that might not have serial numbers, yet are expected to be represented in the EntityTable, include flash cards and power supply modules.

The CCAP Core SHOULD provide the unique component serial number, via the SerialNum object, contained for each FRU that is a pluggable optical module such as an SFP, SFP+, QSFP, XFP, CXP.

The CCAP Core SHOULD provide the unique component serial number, via the SerialNum object, contained within the row entry in the EntityTable for every FRU that is capable of causing and/or generating an event, message, log, or alarm.

7.1.18.12 MfgName

The name of the manufacturer of this physical component. The preferred value is the manufacturer name string actually printed on the component itself (if present).

If the manufacturer name string associated with the physical component is unknown to the agent, then this object will contain a zero-length string.

7.1.18.13 ModelName

The vendor-specific model name identifier string associated with this physical component. The preferred value is the customer-visible part number, which may be printed on the component itself.

If the model name string associated with the physical component is unknown to the agent, then this object will contain a zero-length string.

7.1.18.14 Alias

This object is an 'alias' name for the physical entity, as specified by a network manager, and provides a non-volatile 'handle' for the physical entity.

If the Alias string associated with the physical component is not set, then this object will contain a zero-length string.

7.1.18.15 AssetId

This object is an NMS-assigned asset tracking identifier for the physical entity, and provides non-volatile storage of this information.

If the AssetId string associated with the physical component is not set, then this object will contain a zero-length string.

7.1.18.16 IsFRU

This object indicates whether or not this physical entity is considered a 'field replaceable unit' by the vendor. If this object contains the value 'true(1)' then this entry identifies a field replaceable unit. For all entries that represent components permanently contained within a field replaceable unit, the value 'false(2)' should be returned for this object.

7.1.18.17 MfgDate

This object contains the date of manufacturing of the managed entity. If the manufacturing date is unknown or not supported, the object is not instantiated. The special value '0000000000000000'H may also be returned in this case.

7.1.18.18 Uris

This object contains additional identification information about the physical entity. The object contains URIs and, therefore, the syntax of this object conforms to [RFC 3986], section 2.

Multiple URIs may be present and are separated by white space characters. Leading and trailing white space characters are ignored.

If no additional identification information is known about the physical entity or supported, the object is not instantiated. A zero length octet string may also be returned in this case.

7.1.19 SensorDetails

This table contains one row per physical sensor represented by an associated row in the EntityTable.

An entry in this table describes the present reading of a sensor, the measurement units and scale, and sensor operational status.

Entries are created in this table by the agent. An entry for each physical sensor SHOULD be created at the same time as the associated EntityTable entry. An entry SHOULD be destroyed if the associated EntityTable entry is destroyed.

This object is based on the entPhySensorTable object specified in the ENTITY-SENSOR-MIB [RFC 3433].

Table 7-38 - SensorDetails Object

Attribute Name	Type	Access	Type Constraints	Units
SensorType	enum	Read-Only	other(1), unknown(2), voltsAC(3), voltsDC(4), amperes(5), watts(6), hertz(7), celsius(8), percentRH(9), rpm(10), cmm(11), truthvalue(12)	N/A
Scale	enum	Read-Only	yocto(1), zepto(2), atto(3), femto(4), pico(5), nano(6), micro(7), milli(8), units(9), kilo(10), mega(11), giga(12), tera(13), exa(14), peta(15), zetta(16), yotta(17)	N/A
Precision	unsignedInt	Read-Only		N/A
Value	integer	Read-Only		N/A
OperStatus	enum	Read-Only	ok(1), unavailable(2), nonoperational(3)	N/A
UnitsDisplay	string	Read-Only		N/A
ValueTimeStamp	TimeStamp	Read-Only		N/A
ValueUpdateRate	unsignedInt	Read-Only		milliseconds

7.1.19.1 SensorType

The type of data returned by the associated Value object.

This object SHOULD be set by the agent during entry creation, and the value SHOULD NOT change during operation.

A value of 'other' indicates a measure other than those listed below.

A value of 'unknown' indicates an unknown measurement, or arbitrary, relative numbers.

A value of 'voltsAC' indicates an electric potential.

A value of 'voltsDC' indicates an electric potential.

A value of 'amperes' indicates electric current.

A value of 'watts indicates power.

A value of 'hertz' indicates frequency.

A value of 'celsius' indicates temperature.

A value of 'percentRH' indicates percent relative humidity.

A value of 'rpm' indicates shaft revolutions per minute.

A value of 'cmm' indicates cubic meters per minute (airflow).

A value of 'truthvalue' indicates value returns true(1) or false(2).

7.1.19.2 Scale

This object represents a data scaling factor, represented with an International System of Units (SI) prefix. The actual data units are determined by examining an object of this type together with the associated EntitySensorDataType object.

An object of this type SHOULD be defined together with objects of type EntitySensorDataType and EntitySensorPrecision. Together, associated objects of these three types are used to identify the semantics of an object of type Value.

7.1.19.3 Precision

The number of decimal places of precision in fixed-point sensor values returned by the associated Value object.

This object should be set to '0' when the associated SensorType value is not a fixed-point type: e.g., 'percentRH(9)', 'rpm(10)', 'cmm(11)', or 'truthvalue(12)'.

7.1.19.4 Value

The most recent measurement obtained by the agent for this sensor.

7.1.19.5 OperStatus

The operational status of the sensor.

7.1.19.6 UnitsDisplay

A textual description of the data units that should be used in the display of Value.

7.1.19.7 ValueTimeStamp

The value of sysUpTime at the time the status and/or value of this sensor was last obtained by the agent.

7.1.19.8 ValueUpdateRate

An indication of the frequency that the agent updates the associated Value object, represented in milliseconds.

A value zero indicates:

- the sensor value is updated on demand (e.g., when polled by the agent for a get-request),
- the sensor value is updated when the sensor value changes (event-driven),
- the agent does not know the update rate.

7.1.20 CcapCore

This object is only supported by the CCAP Core and is used to report on the active sessions on the CCAP Core. It has no attributes and the following associations.

Table 7-39 - CcapCore Object Associations

Associated Object Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
SessionInfo	Directed association to SessionInfo	1	0..*	

7.2 Future Additions to the R-PHY MIB

Objects will be added to the R-PHY MIB in a later version of this specification for the following:

- Tracking L2TPv3 session errors on the RPD
- Optical statistics
- Physical security
- Jitter reporting

8 ACCOUNTING MANAGEMENT

There are no additional accounting management requirements to support MHAv2.

9 FAULT MANAGEMENT AND REPORTING REQUIREMENTS

9.1 Fault Management Requirements and Transport Protocols

This section defines requirements for remote monitoring/detection, diagnosis, reporting, and correction of problems.

9.2 Event Reporting[31]

The CCAP Core MUST log events using standard mechanisms defined in section 9 of [CCAP-OSSIv3.1].

The CCAP Core MUST support all Mandatory ("M") CMTS MIB objects that have an SNMP access type of accessible for SNMP Notifications ("Acc-FN") in Annex A of [OSSIv3.0] and Annex A of [L2VPN].

The CCAP Core MUST log events when loss of fan, loss of power supply, and temperature issues are detected. These events are specified in Annex A. The CCAP Core is expected to implement additional physical and environmental events beyond the three basic ones listed here.

Refer to Section 9.2.3 for RPD event reporting.

9.2.1 SNMP Usage

In the DOCSIS environment, SNMP is one method is used to achieve the goals of fault management: remote detection, diagnosis, reporting, and correction of CMTS/CCAP network faults.

The CMTS/CCAP sends SNMP notifications to one or more NMSs (subject to operator imposed policy). CMTS/CCAP requirements for SNMP notifications are detailed in Section 9.2.2.1.2. The CMTS/CCAP sends events to a syslog server. The CMTS/CCAP requirements for syslog events are detailed in Section 9.2.2.1.3.

9.2.2 CCAP Core Event Notification

The CMTS/CCAP generates asynchronous events that indicate malfunction situations and notify the operator about important events. The methods for reporting events are defined below:

1. Stored in Local Log (docsDevEventTable from [RFC 4639]).
2. Reported to SNMP entities as an SNMP notification.
3. Sent as a message to a Syslog server.
4. Optionally reported to NETCONF clients as a NETCONF notification.

This specification defines the support of DOCSIS specific events (see Annex B) and IETF events. The former are normally in the form of SNMP notifications. The delivery of IETF Notifications to local log and syslog server is optional.

Event Notifications are enabled and disabled via configuration settings.

Events can be reported to Local Log, Syslog, and/or SNMP notifications based on the configuration settings defined in the EventReportingCfg object (see Section 6.4.3.6).

The CMTS and CCAP MUST support event notifications via local event logging.

The CMTS and CCAP MUST support event notifications via Syslog, including limiting/throttling, as specified in [RFC 4639].

The CMTS and CCAP MUST support event notification via SNMP traps, including limiting/throttling, as specified in [RFC 4639].

[31] Modified per R-OSSI-N-16.1448-2 on 4/27/16 by KB.

9.2.2.1 Format of Events

The subsections which follow explain in detail how the CMTS and CCAP report standard events by any of the following three mechanisms: local event logging, SNMP notification, and Syslog.

Annex B lists all DOCSIS event definitions.

9.2.2.1.1 Local Event Logging

The CCAP MUST maintain Local Log events, defined in [RFC 4639], in local non-volatile storage.

The CMTS and CCAP MAY retain events designated for local volatile storage in local non-volatile storage.

The CCAP Local Log non-volatile storage events MUST persist across reboots.

Events are identical if their EventIds are identical. For identical events occurring consecutively, the CMTS and CCAP MAY choose to store only a single event.

If the CCAP stores as a single event multiple identical events that occur consecutively, the CCAP MUST reflect the most recent event in the event description.

A CMTS MUST maintain Local Log events, defined in Annex B, in local-volatile storage or local non-volatile storage or both. A CMTS MAY retain in local non-volatile storage events designated for local volatile storage.

A CMTS MUST implement its Local Log as a cyclic buffer. The number of entries supported by the CMTS for the Local Log is vendor-specific with a minimum of ten entries. The CMTS Local Log MAY persist across reboots. The CMTS MUST provide access to the Local Log events through the docsDevEventTable [RFC 4639].

Aside from the procedures defined in this document, event recording conforms to the requirements of [RFC 4639]. Event descriptions are defined in English. A CMTS MUST implement event descriptors such that no event descriptor is longer than 255 characters, which is the maximum defined for SnmpAdminString [RFC 3411].

The EventId digit is a 32-bit unsigned integer. EventIds ranging [RFC 4639] from 0 to (2^{31} - 1) are reserved by DOCSIS. The CMTS MUST report in the docsDevEvTable [RFC 4639] the EventId as a 32-bit unsigned integer and convert the EventId from the error codes defined in Annex B to be consistent with this number format.

The CMTS MUST implement EventIds ranging from 2^{31} to (2^{32} - 1) as vendor-specific EventIds using the following format:

Bit 31 is set to indicate vendor-specific event.

Bits 30-16 contain the lower 15 bits of the vendor's SNMP enterprise number.

Bits 15-0 are used by the vendor to number events.

Section 9.2.2.1.3 describes rules to generate unique EventIds from the error code.

The [RFC 4639] docsDevEvIndex object provides relative ordering of events in the log. The creation of local-volatile and local non-volatile logs necessitates a method for synchronizing docsDevEvIndex values between the two Local Logs after reboot. A CMTS which supports local non-volatile storage MUST adhere to the rules listed below for creating local volatile and local non-volatile logs following a re-boot:

Renumber the values of docsDevEvIndex maintained in the local non-volatile log beginning with 1.

Initialize the local volatile log with the contents of the local non-volatile log.

Use the value of the last restored non-volatile docsDevEvIndex plus one as the docsDevEvIndex for the first event recorded in the new active session's local volatile log.

The CMTS MUST clear both the local volatile and local non-volatile event logs when an event log reset is initiated through an SNMP SET of the docsDevEvControl object [RFC 4639].

9.2.2.1.2 SNMP Notifications

The CCAP MUST implement the generic SNMP notifications according to Annex A.

When any event causes a generic SNMP notification occurrence in a CMTS, the CMTS MUST send notifications if throttling/limiting mechanism [RFC 4639] and other limitations [RFC 3413] do not restrict notification sending.

The CCAP MUST implement SNMP notifications defined in [DOCS-DIAG-MIB] and [DOCS-IF3-MIB].

The CCAP MUST support at least 4 SNMP trap destinations.

The CCAP MUST support the ability to filter traps individually and filter traps by priority level.

A CMTS operating in SNMP v1/v2c NmAccess mode MUST support SNMPv1 and SNMPv2c Traps as defined in [RFC 3416].

A CMTS operating in SNMP Coexistence mode MUST support SNMP notification type 'trap' and 'inform' as defined in [RFC 3416] and [RFC 3413].

The CMTS MUST send notifications for any event, if docsDevEvControl object [RFC 4639], throttling/limiting mechanism [RFC 4639] and [RFC 3413] limitations applied later do not restrict notification sending.

The CMTS MUST NOT report via SNMP notifications vendor-specific events that are not described in instructions submitted with certification testing application documentation.

9.2.2.1.3 Syslog

The CCAP MUST support at least four Syslog servers as recipients.

The CMTS and CCAP MUST support Syslog messages that communicate interface up/down events, user login/logout events, configuration changes, and access failures.

When the CCAP sends a Syslog message for a DOCSIS-defined event, the CCAP MUST send it in the following format:

```
<level>TIMESTAMP HOSTNAME CCAP[vendor]: <eventId> text vendor-specific-text
```

When the CMTS sends a syslog message for a DOCSIS-defined event, the CMTS MUST send it in the following format:

```
<level>TIMESTAMP HOSTNAME CMTS[vendor]: <eventId> text vendor-specific-text
```

Where:

- *level* is an ASCII representation of the event priority, enclosed in angle brackets, which is constructed as an OR of the default Facility (128) and event priority (0-7). The resulting level ranges between 128 and 135.
- *TIMESTAMP* and *HOSTNAME* follow the format of [RFC 3164]. The single space after *TIMESTAMP* is part of the *TIMESTAMP* field. The single space after *HOSTNAME* is part of the *HOSTNAME* field.
- *vendor* is the vendor name for the vendor-specific syslog messages or DOCSIS for the standard DOCSIS messages.
- *eventId* is an ASCII representation of the INTEGER number in decimal format, enclosed in angle brackets, which uniquely identifies the type of event. The CMTS and CCAP MUST equate the eventId with the value stored in the docsDevEvId object in docsDevEventTable. For the standard DOCSIS events this number is converted from the error code using the following rules:
    - The number is an eight-digit decimal number.
    - The first two digits (left-most) are the ASCII code for the letter in the Error code.
    - The next four digits are filled by 2 or 3 digits between the letter and the dot in the Error code with zero filling in the gap in the left side.
    - The last two digits are filled by the number after the dot in the Error code with zero filling in the gap in the left side.

For example, event D04.2 is converted into 68000402, and Event I114.1 is converted into 73011401. This convention only uses a small portion of available number space reserved for DOCSIS (0 to 2^{31}-1). The first letter of an error code is always in upper-case. See Annex B for event definitions.

- *text* contains the textual description for the standard DOCSIS event message, as defined in Annex B.
- *vendor-specific-text* contains vendor-specific information. This field is optional.

For example, the syslog event for the event D04.2, "ToD Response received - Invalid data format", is as follows:

```
<132>CABLEMODEM[DOCSIS]: <68000402> ToD Response received - Invalid data format
```

The number 68000402 in the example is the number assigned by DOCSIS to this particular event.

The CMTS and CCAP MAY report non-DOCSIS events in the standard syslog message format [RFC 3164] rather than the DOCSIS syslog message format defined above.

When the CMTS or CCAP sends a syslog message for an event not defined in this specification, the CMTS or CCAP MAY send it according to the format and semantics of the elements defined above.

9.2.2.2 BIT Values for docsDevEvReporting [RFC 4639]

Permissible BIT values for [RFC 4639] docsDevEvReporting objects include:

1: local(0)

2: traps(1)

3: syslog(2)

4: localVolatile(8)

5: stdInterface(9)

Bit-0 means non-volatile Local Log storage and bit-8 is used for volatile Local Log storage (see Section 9.2.2.1). Bit-1 means SNMP Notifications which correspond to both SNMP Trap and SNMP Inform.

For backward compatibility with Pre-3.0 DOCSIS devices, the CMTS MUST support bit-3 in docsDevEvReporting BITS encoding for volatile Local Log storage.

DOCSIS 3.0 devices need to support bit override mechanisms during SNMP SET operations with either one-byte or two-byte BITS encoding for docsDevEvReporting for backward compatibility with Pre-3.0 DOCSIS behavior.

The CMTS MUST use the bit-3 value to set both bit-3 and bit-8 for SNMP SET operations on docsDevEvReporting using a one-byte BITS encoded value; therefore, the CMTS reports bit-3 and bit-8 with identical values for SNMP GET operations.

The CMTS MUST use the bit-8 value to set bit-3 and bit-8 for SNMP SET operations, irrespective of the bit-3 value, on docsDevEvReporting using a two or more byte BITS encoded value.

The CMTS MAY support bit-9 in docsDevEvReporting BITS encoding in accordance with [RFC 4639] definition.

A CMTS that reports an event by SNMP Notification or syslog MUST also report the event by a Local Log (volatile or non-volatile).

Combinations of docsDevEvReporting with traps(1) and/or syslog(2) bits with no Local Log bits (bit-0, bit-3 or bit-8) set are known as unacceptable combinations.

The CMTS MUST reject and report a 'Wrong Value' error for SNMPv2c/v3 PDUs or a 'Bad Value' error for SNMPv1 PDUs for any attempt to set docsDevEvReporting with unacceptable combinations.

The CMTS MUST accept any SNMP SET operation to docsDevEvReporting different than the unacceptable combinations.

The CMTS MUST ignore any undefined bits in docsDevEvReporting on SNMP SET operations and report a zero value for those bits.

Refer to Section 9.2.2.1.1 for details on Local Log requirements for the CMTS.

If CMTS supports both volatile and non-volatile storage, the CMTS MUST maintain the non-volatile storage when both non-volatile Local Log and volatile Local Log bits are set for a specific docsDevEvReporting event priority. If CMTS supports both volatile and non-volatile storage, the CMTS MAY maintain the volatile storage when both non-volatile Local Log and volatile Local Log bits are set for a specific docsDevEvReporting event priority. When both non-volatile Local Log and volatile Local Log bits are set for a specific docsDevEvReporting event priority, the CMTS MUST NOT report duplicate events in the docsDevEventTable.

9.2.2.3 Standard Events for CCAP

The CCAP MUST maintain the non-volatile storage when both non-volatile Local Log and volatile Local Log bits are set for a specific event priority, configured in the Reporting attribute of the EventReportingCfg object (see Section 6.4.3.6).

The CCAP MAY maintain the volatile storage when both non-volatile Local Log and volatile Local Log bits are set for a specific event priority.

When both non-volatile Local Log and volatile Local Log bits are set for a specific event priority, the CCAP MUST report the event as a single event in the docsDevEventTable.

Event priority levels for the CCAP use the following categories:

Emergency(1) events indicate fatal hardware or software failure that prevent normal system operation (all services are affected).

Alert(2) events indicate a major hardware or software failure that causes some service interruption (no redundancy available).

Critical(3) events indicate a major hardware or software failure that does not cause an interrupt of the normal data flow. This level of event may be also used when some redundant device was automatically activated to replace the defective device.

Error(4) events indicate that an incorrect input signal (external system error) is causing temporary or permanent interruption of the normal data flow.

Warning(5) events indicate a minor failure that does not cause any interrupt of the data flow.

Notice(6) events indicate that a specified alarm condition has been removed.

Information(7) events indicate a milestone or checkpoint in normal operation that could be of particular importance for troubleshooting.

Debug(8) events are reserved for vendor-specific events.

The reporting mechanism for each priority can be changed from the default reporting mechanism via the EventReportingCfg object defined in this specification (see Section 6.4.3.6).

9.2.2.4 Standard DOCSIS Events for CMTS

CMTSs use the same levels of the event priorities as a CM (see [CCAP-OSSIv3.1]); however, the priority definition of the events is different. Events with the priority level of 'Warning' and less, specify problems that could affect the individual user (for example, individual CM registration problem).

Every CMTS vendor may define their own set of 'Alert' events.

Priority level of 'Error' indicates problems with a group of CMs (for example CMs that share same upstream channel).

Priority level of 'Critical' indicates a problem that affects the whole cable system operation, but is not a faulty condition of the CMTS device.

Priority level of 'Emergency' is vendor-specific and indicates problems with the CMTS hardware or software, which prevents CMTS operation.

During CMTS initialization or reinitialization, the CMTS MUST support, as a minimum, the default event reporting mechanism shown in Table 9-1 or Table 9-2 or Table 9-3.

The CMTS MAY implement default reporting mechanisms above the minimum requirements listed in Table 9-1 or Table 9-2 or Table 9-3 with the exception of the 'Debug' priority level.

The reporting mechanism for each priority could be changed from the default reporting mechanism by using docsDevEvReporting object of DOCS-CABLE-DEVICE-MIB [RFC 4639].

Table 9-1 - CMTS Default Event Reporting Mechanism Versus Priority (Non-volatile Local Log Support Only)

Event Priority	Local Log Non-volatile	SNMP Notification	Syslog	Local Log Volatile
Emergency	Yes	No	No	Not Used
Alert	Yes	No	No	Not Used
Critical	Yes	Yes	Yes	Not Used
Error	Yes	Yes	Yes	Not Used
Warning	Yes	Yes	Yes	Not Used
Notice	Yes	Yes	Yes	Not Used
Informational	No	No	No	Not Used
Debug	No	No	No	Not Used

Table 9-2 - CMTS Default Event Reporting Mechanism Versus Priority (Volatile Local Log Support Only)

Event Priority	Local Log Non-volatile	SNMP Notification	Syslog	Local Log Volatile
Emergency	Not Used	No	No	Yes
Alert	Not Used	No	No	Yes
Critical	Not Used	Yes	Yes	Yes
Error	Not Used	Yes	Yes	Yes
Warning	Not Used	Yes	Yes	Yes
Notice	Not Used	Yes	Yes	Yes
Informational	Not Used	No	No	No
Debug	Not Used	No	No	No

Table 9-3 - CMTS Default Event Reporting Mechanism Versus Priority

Event Priority	Local Log Non-volatile	SNMP Notification	Syslog	Local Log Volatile
Emergency	Yes	No	No	No
Alert	Yes	No	No	No
Critical	Yes	Yes	Yes	No
Error	No	Yes	Yes	Yes
Warning	No	Yes	Yes	Yes
Notice	No	Yes	Yes	Yes
Informational	No	No	No	No
Debug	No	No	No	No

The CMTS MUST format notifications for standard DOCSIS events as specified in Annex B.

9.2.3 RPD Event Reporting[32]

An RPD is required to log events and generate asynchronous notifications that indicate malfunction situations and notify the operator about important events. This specification defines a single mechanism for the purpose of RPD event reporting. The RPD MUST report event notifications to the Principal CCAP Core via GCP Notify messages.

The Principal Core maintains the responsibility of reporting events originating from the RPD by methods defined in section 9.2.2. of the CCAP OSSI specification or through vendor proprietary methods such as the Command Line Interface.

9.2.3.1 Format of Events[33]

Annex B lists all DOCSIS events applicable to an RPD.

The following sections explain in detail how to report these events via the local event logging mechanism.

9.2.3.1.1 Local Event Logging

An RPD MUST maintain Local Log events, defined in Annex B, in both local-volatile storage and local non-volatile storage. An RPD MAY retain in local non-volatile storage events designated for local volatile storage. An RPD MAY retain in local volatile storage events designated for local non-volatile storage.

An RPD MUST implement its Local Log as a cyclic buffer with a minimum of ten entries. The RPD Local Log non-volatile storage events MUST persist across reboots. The RPD MUST persist undelivered event reports across reboots.

When the RPD establishes connection to the Principal CCAP Core, the RPD MUST provide all persistent undelivered events for which severity level is enabled based on a request from the CCAP Core. When the RPD establishes connection to the Principal CCAP Core, the RPD MUST NOT forward any event for which severity level is disabled. The RPD allows the Principal CCAP Core to read the cyclic event report buffer upon establishment or re-establishment of the GCP connection. Such approach is intended to enable reporting of events generated during RPD initialization or events related to GCP connectivity.

Aside from the procedures defined in this document, event recording conforms to the requirements of [RFC 4639]. Event descriptions are defined in English. An RPD MUST implement event descriptors such that no event descriptor is longer than 255 characters, which is the maximum defined for SnmpAdminString [RFC 3411].

Events are identical if their EventIds are identical. For identical events occurring consecutively, the RPD MAY choose to store only a single event. If an RPD stores as a single event multiple identical events that occur consecutively, the RPD MUST reflect in the event description the most recent event.

The EventId digit is a 32-bit unsigned integer. EventIds ranging [RFC 4639] from 0 to (2^{31} - 1) are reserved by DOCSIS. The RPD MUST report the EventId as a 32-bit unsigned integer and convert the EventId from the error codes defined in Annex B to be consistent with this number format.

An RPD MUST implement EventIds ranging from 2^{31} to (2^{32} - 1) as vendor-specific EventIds using the following format:

- Bit 31 is set to indicate vendor-specific event
- Bits 30-16 contain the lower 15 bits of the vendor's SNMP enterprise number
- Bits 15-0 are used by the vendor to number events

The RPD MUST adhere to the rules listed below for creating local volatile and local non-volatile logs following a re-boot.

The RPD MUST clear both the local volatile and local non-volatile event logs when an event log reset is initiated through GCP.

[32] Modified per R-OSSI-N-16.1448-2 on 4/27/16 by KB.
[33] Added per R-OSSI-N-16.1448-2 on 4/27/16 by KB.

9.2.3.1.2 Standard DOCSIS Events for RPD

The DOCS-CABLE-DEVICE-MIB [RFC 4639] defines 8 priority levels and a corresponding reporting mechanism for each level.

Emergency event (priority 1)

Reserved for vendor-specific 'fatal' hardware or software errors that prevent normal system operation and cause the reporting system to reboot.

Every vendor may define their own set of emergency events. Examples of such events might be 'no memory buffers available', 'memory test failure', etc.

Alert event (priority 2)

A serious failure, which causes the reporting system to reboot, but it is not caused by hardware or software malfunctioning.

Critical event (priority 3)

A serious failure that requires attention and prevents the device from transmitting data, but could be recovered without rebooting the system. Examples of such events might be the inability to get an IP address from the DHCP server.

Error event (priority 4)

A failure occurred that could interrupt the normal data flow, but will not cause the RPD to re-initialize. Error events could be reported in real time by using GCP.

Warning event (priority 5)

A failure occurred that could interrupt the normal data flow, but will not cause the RPD to re-initialize.

Notice event (priority 6)

The event is important, but is not a failure and could be reported in real time by using GCP. For an RPD, an example of a Notice event is any event from 'SW UPGRADE SUCCESS' group.

Informational event (priority 7)

The event is of marginal importance, and is not failure, but could be helpful for tracing the normal operation.

Debug event (priority 8)

Reserved for vendor-specific non-critical events.

During RPD initialization or reinitialization, the RPD MUST support, as a minimum, the default event reporting mechanism shown in Table 9-4.

The RPD MAY implement default reporting mechanisms above the minimum requirements listed in Table 9-4.

The reporting mechanism for each priority could be changed from the default reporting mechanism by using GCP.

Table 9-4 - RPD default event reporting mechanism versus priority

Event Priority	Local Log Non-volatile	Local Log Volatile
Emergency	Yes	Yes
Alert	Yes	Yes
Critical	Yes	Yes
Error	No	Yes
Warning	No	Yes
Notice	No	Yes
Informational	No	No

Event Priority	Local Log Non-volatile	Local Log Volatile
Debug	No	No

The RPD MUST format notifications that it generates for standard DOCSIS events as specified in Annex B.

9.2.4 Event Priorities and Vendor-Specific Events[34]

This specification defines events that make use of a sub-set of the Event Priority Levels. Vendor-specific events can be defined for any Event Priority Level. Table 9-5 summarizes those considerations.

A CCAP Core MUST assign DOCSIS and vendor specific events as indicated in Table 9-5.

An RPD MUST assign DOCSIS and vendor-specific events as indicated in Table 9-5.

Table 9-5 - Event Priorities Assignment

Event Priority	CCAP Core and RPD Event Assignment
Emergency	Vendor-specific
Alert	CCAP Core and RPD and Vendor-specific (optional*)
Critical	CCAP Core and RPD and Vendor-specific (optional*)
Error	CCAP Core and RPD and Vendor-specific (optional*)
Warning	CCAP Core and RPD and Vendor-specific (optional*)
Notice	CCAP Core and RPD and Vendor-specific (optional*)
Information	CCAP Core and RPD and Vendor-specific (optional*)
Debug	Vendor-specific
* Vendor-specific optional event definitions are recommended only where the CCAP Core or RPD allows for sufficient storage of such events.	

9.2.5 NETCONF Notifications

NETCONF Notifications is an optional mechanism that provides an asynchronous notification message service built on top of the base NETCONF protocol. The mechanism is based on the concept of clients subscribing to events belonging to named event streams. Clients can associate filter parameters with the subscriptions to receive a defined subset of all events belonging to a stream.

Notification replay is an integral part of the NETCONF Notifications framework. It provides the ability for clients to request sending (or resending) recently generated notifications based on a specific start and an optional stop time. If no stop time is provided, the notification stream will continue until the subscription is terminated.

The CCAP MAY implement NETCONF Notifications towards OSS.

If the CCAP implements NETCONF Notifications towards OSS, the CCAP MUST use the YANG module specified for this purpose in [CCAP-EVENTS-YANG].

9.2.6 Trap and Syslog Throttling, Limiting and Inhibiting

A CMTS MUST support SNMP TRAP/INFORM and syslog throttling and limiting as described in DOCS-CABLE-DEVICE-MIB [RFC 4639], regardless of SNMP mode.

9.2.7 Non-SNMP Fault Management Protocols

The OSS can use a variety of tools and techniques to examine faults at multiple layers. For the IP layer, useful non-SNMP based tools include ping (ICMP Echo and Echo Reply), and trace route (UDP and various ICMP Destination

[34] Modified by R-OSSI-N-16.1448-2 on 4/27/16 by KB.

Unreachable flavors). The CMTS MUST support IP end-station generation of ICMP error messages and processing of all ICMP messages.

Syslog requirements are defined in Section 9.2.2.1.3.

9.3 Fault Management UML Object Model

9.3.1 Event Notification Objects

The objects for CCAP Event Notification are derived from the docsDevEventTable in [RFC 4639] and are used without modification.

9.4 RPD Diagnostic LED Indicators[35]

9.4.1 RPD Diagnostic LED Indicators

The RPD implements diagnostic LEDs which denote that a connection has been established with the CCAP Core and indicate the status of the upstream and downstream RF interfaces. Diagnostic LEDs do not replace detailed remote management and monitoring tools, but merely provide a coarse-grained indication of device connectivity to field technicians during their installation or repair of the RPD.

9.4.2 System LED Indicator

The RPD will have a single LED labeled with "System" or "Sys". The System LED will commence flashing at a 2 Hz "slow" rate as soon as the RPD receives power and the RPD CPU has booted. The slow flash rate denotes that the RPD is attempting to communicate across the CIN interface. The LED will begin to flash at the 4 Hz "fast" rate when the RPD has completed IP address acquisition through DHCP and is in the process of IEEE 1588 synchronization and/or being configured by the CCAP Core. At this point, the RPD will be able to accept a remote terminal session (over SSHv2) to perform debugging operations, if necessary. Finally, when the RPD completes provisioning and reaches a fully functional state with respect to the principal CCAP Core, the System LED will steadily illuminate. In addition to the two flash rates and a solid illumination pattern, the LED provides a color pattern to indicate categories of faults. See Table 9-6 for more details.

Table 9-6 - System LEDs

LED Pattern or Color	State Indicated	Notes
Green - 2 Hz Flash Pattern	Power-on - Initializing	Required - Indicates power good
Green - 4 Hz Flash Pattern	IP acquired via DHCP, synchronizing and acquiring configuration	Required - Indicates remote SSHv2 session possible
Green - Steady	System status good - Connections good	No faults detected
Red - 2 Hz Flash Pattern	Connection fault - No physical connection to CIN or RF interfaces	Loose connection, connector faults, water ingress at connector, wavelength mismatch, etc.
Red - 4 Hz Flash Pattern	Configuration fault - Failed synchronization or failed to receive configuration from Principal CCAP Core	Wrong DHCP configuration, unreachable hosts, failed authentication with CCAP Core, etc.
Red - Steady	System fault - Device cannot initialize	Indicates hardware fault, always indicated by a steadily illuminated LED.
Amber - Various	Warning - RPD operational but operation is imperfect	Vendor-defined LED behavior. May indicate high error rates in CIN or RF ports, marginal operating conditions, etc.

9.4.3 CIN LEDs

The RPD will have one LED for each of the interfaces that comprise the CIN. Individual Ethernet ports are identified with "Eth" appended with the numbers 0-n. The LED for an inactive Ethernet interface will not be

[35] Section added per R-OSSI-N-16.1475-2 on 4/27/16 by KB.

illuminated. An inactive interface could be in this state either because the device has been administratively disabled or because it has failed; in both instances, the inactive state of the interface will be indicated by its associated LED remaining unilluminated. The LED flash pattern conforms with [IEEE 802.3x]. See Table 9-7 for more details.

Table 9-7 - CIN LEDs

LED Pattern	State Indicated	Notes
Flash Pattern	Per [IEEE 802.3x]	Indicates connection state
Unilluminated	Ethernet port failure or port administratively disabled	Indicates possible port failure or a port which has been configured to be disabled

9.4.4 RF LEDs

Regardless of the number of physical ports implemented, the RPD implements one LED for the Upstream RF interface labelled "U-RF" and one LED for the Downstream RF interface labelled "D-RF".

The D-RF LED will flash at the 2 Hz rate as soon as the RPD is powered up and the downstream RF output becomes active with at least one pilot tone and/or alignment tone. The D-RF LED will flash at the 4 Hz rate as soon as any configured channel becomes active, and until the RPD receives and applies its complete downstream channel set configuration. Once the downstream channel set configuration has been obtained by the RPD, the D-RF LED will steadily illuminate, indicating the channels are operational. The indication that the downstream channel set configuration is completed is given to the RPD by the principal CCAP Core.

The U-RF LED will flash at the 2 Hz rate, indicating that the RPD has received and applied its upstream channel set configuration. The RPD's U-RF LED will start flashing at the 4 Hz rate as soon as the RPD recognizes the first upstream transmission. The U-RF LED will steadily illuminate when at least one CM completes ranging with the CCAP Core as indicated to the RPD by the CCAP Core.

Additional LEDs might be implemented in the RPD which convey vendor-specific meanings, including power levels, status checks for the various legs of the HFC using addressable elements (such as amplifiers and taps), etc. See Table 9-8 and Table 9-9.

Table 9-8 - D-RF LEDs

D-RF LED Pattern	State Indicated	Notes
2 Hz Flash Pattern	Power-on - Downstream RF initializing	At least one pilot and/or alignment tone active
4 Hz Flash Pattern	Channel or channels activated	Awaiting downstream channel set configuration
Steadily Illuminated	Downstream channel set configuration obtained	Downstream channels operational
Unilluminated	RF port fault or RF port mute	Fault detected in RF port or RF port muted by CCAP Core

Table 9-9 - U-RF LEDs

U-RF LED Pattern	State Indicated	Notes
2 Hz Flash Pattern	At least one upstream channel configuration received (and a MAP was received for that channel), awaiting upstream burst transmissions	Indicates that the RPD has received and applied at least one upstream channel configuration
4 Hz Flash Pattern	First upstream transmission recognized	Indicates at least one CM's burst transmission has been received
Steadily Illuminated	At least one CM has successfully completed upstream ranging	Upstream deemed operational
Unilluminated	Upstream RF port fault or no configuration	Fault detected in US RF port or no configuration received.

10 SNMP AND MIB REQUIREMENTS[36]

Most CCAP MIB objects are used in a read-only mode for status and performance monitoring. The CCAP Core requires a very small set of read-create or read-write MIB objects used by operators for operational control, automation or testing tasks, but since the RPD is bootstrapped and configured via the CCAP, it does not. The RPD is not required to support SNMP or any MIB objects directly - RPD SNMP data is reported through the CCAP Core and conveyed from the RPD via GCP. To the polling entity, an RPD appears to be part of a CCAP Core, similar to the way that a linecard is presented in an integrated CCAP.

10.1 Protocol and Agent Requirements

The CCAP Core MUST meet the SNMP and MIB requirements specified in [CCAP-OSSIv3.1], except where specified differently in this specification.

The CCAP Core MUST support all mandatory MIB objects specified in the Detailed MIB Requirements (Normative) annex of [CCAP-OSSIv3.1], except where specified differently in this specification.

The CCAP Core MUST support all mandatory MIB objects specified in Table A-4 - CCAP Core MIB Object Details.

The RPD MAY support read-only access via the SNMPv2c protocol.

If the RPD supports SNMP, the RPD MAY support the SNMPv3 protocol.

The RPD MUST NOT support write or create SNMP operations.

If the RPD supports SNMP, the RPD MUST meet the SNMP requirements specified in [CCAP-OSSIv3.1], except where specified differently in this specification.

If the RPD supports SNMP, the RPD SHOULD support the MIB objects specified in Table A-3 - RPD MIB Object Details.

10.2 CableLabs MIBs

Table 10-1 - R-PHY CableLabs MIBs

Reference	MIB Module
[DOCS-RPHY-MIB]	To be developed

10.3 Specific MIB Object Implementation Requirements[37]

10.3.1 Requirements for Interfaces Group MIB [RFC 2863]

The CCAP Core MUST implement the interface MIB [RFC 2863].

In addition to the ifTypes defined in [CCAP-OSSIv3.1], the following ifType and enumerated value has been added for R-PHY:

- CATV bi-directional RF port: docsCableBidirRfPort (301)

The following statements define the RPD interface-numbering scheme requirements:

The CCAP Core MUST implement a row entry in the ifTable for each downstream channel, upstream interface, and logical upstream channel that exists in the RPD.

[36] Modified per R-OSSI-N-15.1407-1 and R-OSSI-N-15-1412-4 on 1/12/15 by KB.
[37] Modified per R-OSSI-N-16.1429-1 on 4/22/2016 by KB.

The CCAP Core MUST implement a row entry in the ifTable for each Bidirectional RF Port in the RPD chassis. A Bidirectional RF Port is typically associated with a single F-connector. The CCAP Core MUST implement an ifType value of 301 in the ifTable row entry for each Bidirectional RF Port.

The CCAP Core MUST implement a row entry in the ifTable for each Downstream RF Port in the RPD. A Downstream RF Port is typically associated with a single F-connector. The CCAP Core MUST implement an ifType value of 257 in the ifTable row entry for each Downstream RF Port.

When an instance of VideoDownChannel is created on a given Downstream or Bidirectional RF Port, the CCAP Core MUST implement a row entry in the ifTable with an ifType value of 214 (QAM). For replicated QAMs, an ifTable entry will be created for every instance of a video QAM on a given Downstream or Bidirectional RF Port, regardless of whether the QAM has been replicated.

When an instance of DocsisDownChannel is created on a given Downstream or Bidirectional RF Port, the CCAP Core MUST implement a row entry in the ifTable with an ifType value of 128 (docsCableDownstream).

When an instance of DOCSIS OFDMDownstreamChannel is created on a given Downstream or Bidirectional RF Port, the CCAP Core MUST implement a row entry in the ifTable with an ifType value of 277 (docsOfdmDownstream).

The CCAP Core MUST implement a row entry in the ifTable for each Upstream RF Port in the RPD. An Upstream RF Port is typically associated with a single F-connector. The CCAP Core MUST implement an ifType value of 256 in the ifTable row entry for each Upstream RF Port.

When an instance of DOCSIS UpstreamPhysicalChannel is created on a given Upstream or Bidirectional RF Port, the CCAP Core MUST automatically create one or more corresponding instances of an UpstreamLogicalChannel.

When an instance of DOCSIS UpstreamPhysicalChannel is created on a given Upstream or Bidirectional RF Port, the CCAP Core MUST implement a row entry in the ifTable with an ifType value of 129 (docsCableUpstream).

When an instance of DOCSIS OFDMAUpstreamChannel is created on a given Upstream or Bidirectional RF Port, the CCAP Core MUST implement a row entry in the ifTable with an ifType value of 278 (docsOfdmaUpstream).

When an instance of DOCSIS UpstreamLogicalChannel is created on a given Upstream or Bidirectional RF Port, the CCAP Core MUST implement a row entry in the ifTable with an ifType value of 205 (docsCableUpstreamChannel).

For each loopback interface that is defined in the system, the CCAP Core MUST implement a row entry in the ifTable with an ifType value of 24, per [RFC 2863].

For each row entry created in the ifTable, the CCAP Core MUST create a corresponding row entry in the ifXTable.

The CCAP Core SHOULD maintain the same ifIndex value for configured interfaces across reboots if there have been no configuration changes. The interfaces to be persisted across reboots include those interfaces specified in the RPD configuration UML object model.

10.3.1.1 ifAdminStatus, ifOperStatus and Traffic

The RPD reports ifOperStatus via GCP according to the last CCAP-configured value of ifAdminStatus, or as 'down' for locally detected interface failure.

10.3.1.2 SNMP Notification Control Requirements

If a multi-layer interface model is present in the device, each sub-layer for which there is an entry in the ifTable can generate linkUp/Down traps. Since interface state changes would tend to propagate through the interface stack (from top to bottom, or bottom to top), it is likely that several traps would be generated for each linkUp/Down occurrence. The ifLinkUpDownTrapEnable object allows managers to control SNMP notification generation, and configure only the interface sub-layers of interest.

If the RPD supports SNMP, the RPD MUST NOT transmit link notifications for RF channel interfaces.

The RPD reports link notifications for other interfaces as configured by the CCAP Core. At startup, the RPD MUST initialize the value of ifLinkUpDownTrapEnable to disabled(2). Thereafter, the RPD MUST report the value of ifLinkUpDownTrapEnable as configured by the CCAP Core.

10.3.1.3 ifTable and ifXTable Counters

The CCAP Core MUST implement the ifTable and ifXTable [RFC 2863] Counter32 and Counter64 MIB objects as defined for the bidirectional RF port interface as described in Table 10-3.

10.3.1.4 CCAP Core ifStack Table

Shown below is an example of how the ifStack table might look for RF interfaces on the RPD. The values used for the ifIndexes are for example purposes only. The relationships are consistent with those defined in [CCAP-OSSIv3.1] but also add the Bidirectional RF Port, which is an RPD-only concept.

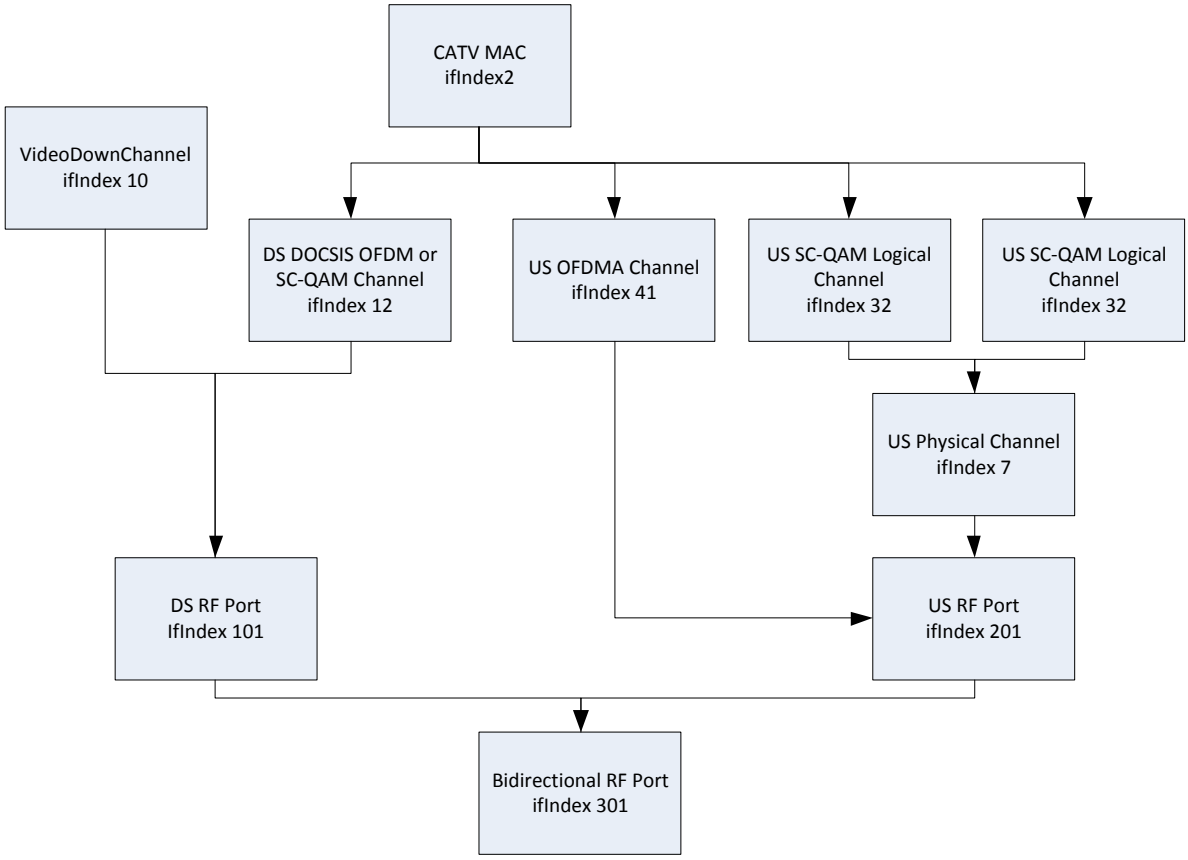

Figure 10-1 - ifStack Table for RPD RF Interfaces

Table 10-2 - CCAP Core ifStack Table Representation[38]

ifStackHigherLayer	ifStackLowerLayer
0	2
0	10
2	12
2	32
2	41

[38] Modified per R-OSSI-N-16.1429-1 on 4/22/16 by KB.

ifStackHigherLayer	ifStackLowerLayer
7	201
10	101
12	101
32	7
41	201
101	301
201	301
301	0

10.3.1.5 IF-MIB Detailed Requirements

Table 10-3 details the specific ifTable and ifXTable MIB values that are expected for the bidirectional RF interfaces on the RPD as reported by the CCAP Core. All other interfaces are as specified in [CCAP-OSSIv3.1].

Table 10-3 - IfTable/IfXTable for Bidirectional RF Interfaces

MIB Objects	BidirRf Port
IfTable	
ifIndex	(n)
ifDescr	
ifType	301
ifMtu For RF Upstream/Downstream; the value includes the length of the MAC header.	0
ifSpeed	0
ifPhysAddress	Empty-String
ifAdminStatus Refer to 7.1.16.8	up(1), down(2), testing(3)
ifOperStatus Refer to 7.1.16.9	up(1), down(2), testing(3), dormant(5), notPresent(6)
ifLastChange	
ifXTable	
ifName	
ifLinkUpDownTrapEnable Refer to 7.1.16.26	
ifHighSpeed	0
ifPromiscuousMode	false(2)
ifConnectorPresent	
ifAlias	
ifCounterDiscontinuityTime	

10.3.2 Requirements for Entity-MIB [RFC 4133]

If the RPD supports SNMP, the RPD MAY implement the ENTITY-MIB [RFC 4133].

The CCAP Core MUST implement the ENTITY-MIB [RFC 4133].

For each row entry created in the SNMPv2-MIB ifTable that can be mapped to an entity represented in the R-PHY-MIB Entity Table, the CCAP Core MUST create a corresponding row entry in the entAliasMappingTable.

10.3.2.1 Guidelines for the implementation of the Entity MIB

The Entity MIB [RFC 4133] provides a physical component layer applicable to managed objects defined for DOCSIS devices. In particular for the entPhysicalTable MIB objects, not all the physical components listed need to instantiate all the object's attributes in entPhysicalTable (the Maximum Access is as defined in [RFC 4133]).

The following table represents high level constraints for any instance of entPhysicalTable.

Table 10-4 - entPhysicalTable Requirements

MIB object	Value
entPhysicalIndex	n
entPhysicalDescr	Text Description
entPhysicalVendorType	Enterprise-specific OID or zeroDotZero
entPhysicalContainedIn	0..n
entPhysicalClass	Physical Class per [RFC 4133]
entPhysicalParentRelPos	-1..n per [RFC 4133]
entPhysicalName	Physical element name In case of a component mapped to an interface Index ifName can be reported, otherwise zero-length string
entPhysicalHardwareRev	Hardware revision or zero-length string
entPhysicalFirmwareRev	Firmware revision or zero-length string
entPhysicalSoftwareRev	Software revision or zero-length string
entPhysicalSerialNum	Serial Number or zero-length string
entPhysicalMfgName	Manufacturer Name or zero-length string
entPhysicalModelName	Model Name or zero-length string
entPhysicalAlias	Physical element operator defined alias In case of a component mapped to an interface Index ifAlias can be reported and implemented as read-only, otherwise zero-length string
entPhysicalAssetID	User defined Asset ID or zero-length string
entPhysicalIsFRU	'true' or 'false'
entPhysicalMfgDate	Manufacturer data or all zeros '0000000000000000'H
entPhysicalUris	URI or zero-length string

10.3.3 Requirements for Entity Sensor MIB [RFC 3433]

The RPD MAY implement the Entity Sensor MIB [RFC 3433].

10.3.4 Requirements for Bridge MIB [RFC 4188]

The requirements for bridging and link-layer forwarding are under consideration and requirements, if needed, will be added to a future version of this specification.

10.3.5 Requirements for Internet Protocol MIB [RFC 4293]

The CCAP Core implements this MIB for the interfaces that are native to the core as specified in [CCAP-OSSIv3.1]. IP interfaces on the RPD are reported by the CCAP Core in the R-PHY MIB. These objects will be defined in a later version of this specification.

10.3.6 Requirements for DOCSIS Remote PHY MIB [DOCS-RPHY-MIB]

The Remote PHY MIB provides details about each RPD to which a CCAP Core is attached. The MIB provides information about the identity and capability of the RPD, the sessions established between the CCAP Core and each RPD, and the interfaces and entities that are contained within each RPD. It also provides the sessions that are available on the CCAP Core.

Not all tables are mandatory for the CCAP Core and RPD. The CCAP Core MUST implement the R-PHY-MIB as described in Section 7 and Annex A. If the RPD supports SNMP, the RPD MUST implement the R-PHY-MIB as described in Section 7 and Annex A.

10.3.7 Requirements for 8021X-PAE MIB [RFC 4022]

If the RDP supports SNMP, the RPD MUST implement the 8021X-PAE-MIB (additional details are expected in a future version of the specification).

11 SECURITY MANAGEMENT[39]

This section defines CCAP Core and RPD requirements for security management functions.

11.1 Secure Shell Requirements[40]

During normal operation, the RPD is managed through the CCAP Core. However, it is anticipated that operators will require secure remote access to the RPD for activities such as set-up prior to installation, maintenance, and troubleshooting. The RPD provides a Secure Shell (SSH) server that allows secure remote access and interaction with the RPD via vendor-specific command line interface.

The RPD MUST support SSH version 2 as defined in:

- [RFC 4250]
- [RFC 4251]
- [RFC 4252]
- [RFC 4253]
- [RFC 4254]
- [RFC 6668]

In addition to the ciphers specified in the SSH RFCs, the RPD MUST support AES-128 as specified in [FIPS-197].

The RPD SHOULD support the following ciphers:

- AES-192 as specified in [FIPS-197]
- AES-256 as specified in [FIPS-197]
- Three-key 3DES in CBC mode as specified in [3DES]

In addition to the MAC algorithms specified in the SSH RFCs, the RPD SHOULD support hmac-sha2-256, specified in [FIPS-180].

Security standards change over time as older algorithms are more easily compromised and new standards are developed. Operators expect an RPD to implement safe, reliable encryption algorithms. The National Institute of Standards and Technology provides recommendations for cryptographic algorithms and key lengths in [SP 800-131]; it is expected that the SSH implementations will take these recommendations into account and update cryptographic capabilities accordingly.

Since an RPD may be installed in an untrusted part of the MSO's network, secure access to the RPD is required. If MACsec is not being used between the RPD and the Network Access Device (NAD), then to avoid the threat of a Man-in-the-Middle (MITM) attack, it is strongly recommended mutual authentication is used within SSH between the RPD and the NMS. If MACsec is used between the RPD and the NAD, then threat for a MITM attack is avoided. In this case, it is not necessary for the NMS SSH client to authenticate the RPD. In either case, the RPD SSH server is always required to authenticate the NMS client.

An RPD may be shipped to an operator with less secure local and remote access than needed for deployment in an Outside Plant (OSP). Therefore, it is important than an RPD is properly secured, prior to installation in the OSP, by performing some pre-staging steps. The pre-staging steps include accessing the RPD's vendor-specific command line interface using one of the less secure access methods, configuring the SSH authorization list with at least one NMS client public key, and disabling all less secure access methods. Optionally, the RPD's public key could be obtained and installed on the NMS client for use during mutual SSH authentication.

This is all captured by the following requirements.

The RPD MUST support mutual authentication within SSH. Note that an operator may choose not to support NMS authentication of RPDs.

[39] Modified per R-OSSI-N-16.1494-1 on 4/22/16 by KB.
[40] Added per R-OSSI-N-15.1411-3 on 1/8/15 by KB.

The RPD's SSH server MUST authenticate an NMS client's public key during establishment of an SSH session with the NMS.

It is strongly recommended that each NMS SSH client authenticates the RPD's public key during establishment of an SSH session with the RPD.

The RPD MUST provide a method to store an NMS client public key into the RPD's authorization list as part of an RPD pre-staging effort.

The RPD MUST be capable of storing a minimum of 16 NMS public keys.

The RPD MUST provide a method to retrieve its public key (derived from the RPD's X.509 Device Certificate) as part of an RPD pre-staging effort.

An RPD vendor could also provide a list of RPD public keys to an operator for all RPDs shipped to the operator.

In this case, the operator would install the public keys of all RPDs into the NMS authorization list used for SSH authentication of RPDs.

11.2 Certificate Management[41]

The CMTS in the CCAP Core has certificate MIBs used to control and view certificate validation functions of Cable Modems. These are defined in docsBpi2CmtsCertObjects of DOCS-IETF-BPI2-MIB [RFC 4131]. The CCAP Core MUST support the docsBpi2CmtsCACertTable of DOCS-IETF-BPI2-MIB [RFC 4131] for RPD certificate validation functions.

The DOCS-BPI2EXT-MIB has certificate MIBs used to control and view certificate validation functions of RPDs. The RPD SHOULD support docsBpi2Ext31RpdDeviceCertTable of [DOCS-BPI2EXT-MIB]. The RPD SHOULD support the docsBpi2Ext31CodeDownloadControl group of MIB objects from [DOCS-BPI2EXT-MIB].

[41] Added per R-OSSI-N-16.1465-2 on 4/25/2016 by KB.

Annex A Detailed MIB Requirements (Normative)

This Annex defines the SNMP MIB modules and MIB variables required for DOCSIS 3.1 CMTS and CCAP devices. Refer to Section 2.1 for the associated MIB files.

Table A-1 - MIB Implementation Support

Requirement Type	Table Notation	Description
Deprecated	D	Deprecated objects are optional. If a vendor chooses to implement the object, the object is expected to be implemented correctly according to the MIB definition. If a vendor chooses not to implement the object, an agent is expected to respond with the appropriate error/exception condition (e.g., 'noSuchObject' for SNMPv2c).
Mandatory	M	The object is expected to be implemented correctly according to the MIB definition.
Not Applicable	NA	Not applicable to the device.
Not Supported	N-Sup	An agent is expected to respond with the appropriate error/exception condition (e.g., 'noSuchObject' for SNMPv2c).
Optional	O	A vendor can choose to implement or not implement the object. If a vendor chooses to implement the object, the object is expected to be implemented correctly according to the MIB definition. If a vendor chooses not to implement the object, an agent is expected to respond with the appropriate error/exception condition (e.g., 'noSuchObject' for SNMPv2c).
Obsolete	Ob	In SNMP convention, obsolete objects should not be implemented. This specification allows vendors to implement or not implement obsolete objects. If a vendor chooses to implement an obsoleted object, the object is expected to be implemented correctly according to the MIB definition. If a vendor chooses not to implement the obsoleted object, the SNMP agent is expected to respond with the appropriate error/exception condition (e.g., 'noSuchObject' for SNMPv2c).

Table A-2 - SNMP Access Requirements

SNMP Access Type	Table Notation	Description
N-Acc	Not Accessible	The object is not accessible and is usually an index in a table
Read Create	RC	The access of the object MUST be implemented as Read-Create
Read Write	RW	The access of the object MUST be implemented as Read-Write
Read Only	RO	The access of the object MUST be implemented as Read-Only
Read Create or Read Only	RC/RO	The access of the object MUST be implemented as either Read-Create or Read-Only as described in the MIB definition
Read Write / Read Only	RW/RO	The access of the object MUST be implemented as either Read-Write or Read-Only as described in the MIB definition
Accessible for SNMP Notifications	Acc-FN	These objects are used for SNMP Notifications by the CMTS and CM SNMP Agents

A.1 RPD MIB Object Details[42]

An RPD optionally supports SNMP. Table A-3 provides the MIB requirements for an RPD that supports SNMP.

Table A-3 - RPD MIB Object Details[43]

BRIDGE-MIB [RFC 4188] **Note: Implementation of BRIDGE-MIB is required ONLY if device is a bridging device.** **This MIB needs to be revisited.**		
SNMPv2-MIB [RFC 3418]		
Object	**RPD**	**Access**

[42] Modified per R-OSSI-N-15.1407-1 and R-OSSI-N-15.1412-4 on 1/11/2015 by KB.
[43] Modified per R-OSSI-N-16.1429-1 and R-OSSI-N-16.1443-2 on 4/25/16 by KB.

Object	RPD	Access
SystemGroup		
sysDescr	M	RO
sysObjectID	M	RO
sysUpTime	M	RO
sysContact	M	RO
sysName	M	RO
sysLocation	M	RO
sysServices	M	RO
sysORLastChange	M	RO
sysORTable	N-SUP	N-Acc
sysOREntry	N-SUP	N-Acc
sysORIndex	N-SUP	N-Acc
sysORID	N-SUP	RO
sysORDescr	N-SUP	RO
sysORUpTime	N-SUP	RO
SNMPGroup	N-SUP	
snmpInPkts	N-SUP	RO
snmpInBadVersions	N-SUP	RO
snmpOutPkts	Ob	RO
snmpInBadCommunityNames	N-SUP	RO
snmpInBadCommunityUses	N-SUP	RO
snmpInASNParseErrs	N-SUP	RO
snmpInTooBigs	Ob	RO
snmpInNoSuchNames	Ob	RO
snmpInBadValues	Ob	RO
snmpInReadOnlys	Ob	RO
snmpInGenErrs	Ob	RO
snmpInTotalReqVars	Ob	RO
snmpInTotalSetVars	Ob	RO
snmpInGetRequests	Ob	RO
snmpInGetNexts	Ob	RO
snmpInSetRequests	Ob	RO
snmpInGetResponses	Ob	RO
snmpInTraps	Ob	RO
snmpOutTooBigs	Ob	RO
snmpOutNoSuchNames	Ob	RO
snmpOutBadValues	Ob	RO

Object	RPD	Access
snmpOutGenErrs	Ob	RO
snmpOutGetRequests	Ob	RO
snmpOutGetNexts	Ob	RO
snmpOutSetRequests	Ob	RO
snmpOutGetResponses	Ob	RO
snmpOutTraps	Ob	RO
snmpEnableAuthenTraps	N-SUP	RO
snmpSilentDrops	N-SUP	RO
snmpProxyDrops	N-SUP	RO
snmpTrapsGroup		
coldStart	N-SUP	Acc-FN
warmStart	N-SUP	Acc-FN
authenticationFailure	N-SUP	Acc-FN
snmpSetGroup		
snmpSetSerialNo	N-SUP	RO
Etherlike-MIB [RFC 3635]		
Object	**RPD**	**Access**
dot3StatsTable	O	N-Acc
dot3StatsEntry	O	N-Acc
dot3StatsIndex	O	RO
dot3StatsAlignmentErrors	O	RO
dot3StatsFCSErrors	O	RO
dot3StatsInternalMacTransmitErrors	O	RO
dot3StatsFrameTooLongs	O	RO
dot3StatsInternalMacReceiveErrors	O	RO
dot3StatsSymbolErrors	O	RO
dot3StatsSingleCollisionFrames	O	RO
dot3StatsMultipleCollisionFrames	O	RO
dot3StatsDeferredTransmissions	O	RO
dot3StatsLateCollisions	O	RO
dot3StatsExcessiveCollisions	O	RO
dot3StatsCarrierSenseErrors	O	RO
dot3StatsDuplexStatus	O	RO
dot3StatsSQETestErrors	N-Sup	
dot3CollTable	O	N-Acc
dot3CollEntry	O	N-Acc
dot3CollCount	O	N-Acc

Object	RPD	Access
dot3CollFrequencies	O	RO
dot3ControlTable	O	N-Acc
dot3ControlEntry	O	N-Acc
dot3ControlFunctionsSupported	O	RO
dot3ControlInUnknownOpcodes	O	RO
dot3PauseTable	O	N-Acc
dot3PauseEntry	O	N-Acc
dot3PauseAdminMode	O	RO
dot3PauseOperMode	O	RO
dot3InPauseFrames	O	RO
dot3OutPauseFrames	O	RO
HOST-RESOURCES-MIB [RFC 2790]		
Object	**RPD**	**Access**
hrDeviceTable	O	N-Acc
hrDeviceEntry	O	N-Acc
hrDeviceIndex	O	RO
hrDeviceType	O	RO
hrDeviceDescr	O	RO
hrDeviceID	O	RO
hrDeviceStatus	O	RO
hrDeviceErrors	O	RO
hrSystem		
hrMemorySize	O	RO
hrStorageTable	O	N-Acc
hrStorageEntry	O	N-Acc
hrStorageIndex	O	RO
hrStorageType	O	RO
hrStorageDescr	O	RO
hrStorageAllocationUnits	O	RO
hrStorageSize	O	RO
hrStorageUsed	O	RO
hrStorageAllocationFailures	O	RO
hrSWRunTable	O	N-Acc
hrSWRunEntry	O	N-Acc
hrSWRunIndex	O	RO
hrSWRunName	O	RO
hrSWRunID	O	RO

Object	RPD	Access
hrSWRunPath	O	RO
hrSWRunParameters	O	RO
hrSWRunType	O	RO
hrSWRunStatus	O	RO
hrSWRunPerfTable	O	N-Acc
hrSWRunPerfEntry	O	N-Acc
hrSWRunPerfCPU	O	RO
hrSWRunPerfMem	O	RO
hrProcessorTable	O	N-Acc
hrProcessorEntry	O	N-Acc
hrProcessorFrwID	O	RO
hrProcessorLoad	O	RO
SNMP-COMMUNITY-MIB [RFC 3584]		
Object	**RPD**	**Access**
snmpCommunityTable	O	N-Acc
snmpCommunityEntry	O	N-Acc
snmpCommunityIndex	O	N-Acc
snmpCommunityName	O	RO
snmpCommunitySecurityName	O	RO
snmpCommunityContextEngineID	O	RO
snmpCommunityContextName	O	RO
snmpCommunityTransportTag	O	RO
snmpCommunityStorageType	O	RO
snmpCommunityStatus	O	RO
snmpTargetAddrExtTable	N-Supp	N-Acc
snmpTargetAddrExtEntry	N-Supp	N-Acc
snmpTargetAddrTMask	N-Supp	RO
snmpTargetAddrMMS	N-Supp	RO
snmpTrapAddress	N-Supp	ACC-FN
snmpTrapCommunity	N-Supp	ACC-FN
BFD-STD-MIB		
Object	**RPD**	**Access**
Additional Details in future version of specification		
IEEE8021X-PAE-MIB		
Object	**RPD**	**Access**
Additional Details in future version of specification		

Object	RPD	Access
DOCS-RPHY-MIB [DOCS-RPHY-MIB]		
Additional details will be provided in a future version of the specification.	**RPD**	**Access**

A.2 CCAP Core MIB Object Details[44]

The CCAP Core is required to support the MIBs detailed in [CCAP-OSSIv3.1]. In addition, the CCAP Core will support the MIBs as detailed in Table A-4.

Table A-4 - CCAP Core MIB Object Details[45]

BFD-STD-MIB		
Object	**CCAP**	**Access**
Additional Details in future version of specification		
IEEE8021X-PAE-MIB		
Object	**CCAP**	**Access**
Additional Details in future version of specification		
IEEE8021-SECY-MIB		
Object	**CCAP**	**Access**
SecY Management		
secyIfTable	M	N-Acc
secyIfEntry	M	N-Acc
secyIfInterfaceIndex	M	N-Acc
secyIfMaxPeerSCs	M	RO
secyIfRxMaxKeys	M	RO
secyIfTxMaxKeys	M	RO
secyIfProtectFramesEnable	M	RW
secyIfValidateFrames	M	RW
secyIfReplayProtectEnable	M	RW
secyIfReplayProtectWindow	M	RW
secyIfCurrentCipherSuite	M	RW
secyIfAdminPt2PtMAC	M	RW
secyIfOperPt2PtMAC	M	RO
secyIfIncludeSCIEnable	M	RW
secyIfUseESEnable	M	RW
secyIfUseSCBEnable	M	RW
Tx SC Management		
secyTxSCTable	M	N-Acc

[44] Added/modified per R-OSSI-N-15.1407-1 and R-OSSI-N-15.1412-4 on 1/11/2015 by KB.
[45] Modified per R-OSSI-N-16.1429-1, R-OSSI-N-16.1443-2, and R-OSSI-N-16.1461-3 on 4/22/16 by KB.

Object	CCAP	Access
secyTxSCEntry	M	N-Acc
secyTxSCI	M	RO
secyTxSCState	M	RO
secyTxSCEncodingSA	M	RO
secyTxSCEncipheringSA	M	RO
secyTxSCCreatedTime	M	RO
secyTxSCStartedTime	M	RO
secyTxSCStoppedTime	M	RO
Tx SA Management		
secyTxSATable	M	N-Acc
secyTxSAEntry	M	N-Acc
secyTxSA	M	N-Acc
secyTxSAState	M	RO
secyTxSANextPN	M	RO
secyTxSAConfidentiality	M	RO
secyTxSASAKUnchanged	M	RO
secyTxSACreatedTime	M	RO
secyTxSAStartedTime	M	RO
secyTxSAStoppedTime	M	RO
Rx SC Management		
secyRxSCTable	M	N-Acc
secyRxSCEntry	M	N-Acc
secyRxSCI	M	N-Acc
secyRxSCState	M	RO
secyRxSCCurrentSA	M	RO
secyRxSCCreatedTime	M	RO
secyRxSCStartedTime	M	RO
secyRxSCStoppedTime	M	RO
Rx SA Management		
secyRxSATable	M	N-Acc
secyRxSAEntry	M	N-Acc
secyRxSA	M	N-Acc
secyRxSAState	M	RO
secyRxSANextPN	M	RO
secyRxSASAKUnchanged	M	RO
secyRxSACreatedTime	M	RO
secyRxSAStartedTime	M	RO

Object	CCAP	Access
secyRxSAStoppedTime	M	RO
SecY Selectable Cipher Suites		
secyCipherSuiteTable	M	N-Acc
secyCipherSuiteEntry	M	N-Acc
secyCipherSuiteIndex	M	N-Acc
secyCipherSuiteId	M	RC
secyCipherSuiteName	M	RC
secyCipherSuiteCapability	M	RC
secyCipherSuiteProtection	M	RC
secyCipherSuiteProtectionOffset	M	RC
secyCipherSuiteDataLengthChange	M	RC
secyCipherSuiteICVLength	M	RC
secyCipherSuiteRowStatus	M	RC
TX SA Statistics		
secyTxSAStatsTable	M	N-Acc
secyTxSAStatsEntry	M	N-Acc
secyTxSAStatsProtectedPkts	M	RO
secyTxSAStatsEncryptedPkts	M	RO
TX SC Statistics		
secyTxSCStatsTable	M	N-Acc
secyTxSCStatsEntry	M	N-Acc
secyTxSCStatsProtectedPkts	M	RO
secyTxSCStatsEncryptedPkts	M	RO
secyTxSCStatsOctetsProtected	M	RO
secyTxSCStatsOctetsEncrypted	M	RO
RX SA Statistics		
secyRxSAStatsTable	M	N-Acc
secyRxSAStatsEntry	M	N-Acc
secyRxSAStatsUnusedSAPkts	M	RO
secyRxSAStatsNoUsingSAPkts	M	RO
secyRxSAStatsNotValidPkts	M	RO
secyRxSAStatsInvalidPkts	M	RO
secyRxSAStatsOKPkts	M	RO
RX SC Statistics		
secyRxSCStatsTable	M	N-Acc
secyRxSCStatsEntry	M	N-Acc
secyRxSCStatsUnusedSAPkts	M	RO

Object	CCAP	Access
secyRxSCStatsNoUsingSAPkts	M	RO
secyRxSCStatsLatePkts	M	RO
secyRxSCStatsNotValidPkts	M	RO
secyRxSCStatsInvalidPkts	M	RO
secyRxSCStatsDelayedPkts	M	RO
secyRxSCStatsUncheckedPkts	M	RO
secyRxSCStatsOKPkts	M	RO
secyRxSCStatsOctetsValidated	M	RO
secyRxSCStatsOctetsDecrypted	M	RO
SECY Statistics		
secyStatsTable	M	N-Acc
secyStatsEntry	M	N-Acc
secyStatsTxUntaggedPkts	M	RO
secyStatsTxTooLongPkts	M	RO
secyStatsRxUntaggedPkts	M	RO
secyStatsRxNoTagPkts	M	RO
secyStatsRxBadTagPkts	M	RO
secyStatsRxUnknownSCIPkts	M	RO
secyStatsRxNoSCIPkts	M	RO
secyStatsRxOverrunPkts	M	RO
DOCS-RPHY-MIB [DOCS-RPHY-MIB]		
Object	**CCAP**	**Access**
docsRphyRpdInfoTable	M	N-Acc
docsRphyRpdInfoEntry	M	N-Acc
docsRphyRpdInfoUniqueId	M	N-Acc
docsRphyRpdInfoSysUpTime	M	RO
docsRphyRpdIdentificationTable	M	N-Acc
docsRphyRpdIdentificationEntry	M	N-Acc
docsRphyRpdIdVendorName	M	RO
docsRphyRpdIdVendorId	M	RO
docsRphyRpdIdModelNum	M	RO
docsRphyRpdIdSerialNum	M	RO
docsRphyRpdIdDeviceAlias	M	RO
docsRphyRpdIdDeviceDescr	M	RO
docsRphyRpdIdHwRev	M	RO
docsRphyRpdIdCurrSwVer	M	RO

Object	CCAP	Access
docsRphyRpdIdBootRomVer	M	RO
docsRphyRpdLocationTable	M	N-Acc
docsRphyRpdLocationEntry	M	N-Acc
docsRphyRpdLocationDescr	M	RO
docsRphyRpdLocationLatitude	M	RO
docsRphyRpdLocationLongitude	M	RO
docsRphyRpdCoresConnectedTable	M	N-Acc
docsRphyRpdCoresConnectedEntry	M	N-Acc
docsRphyRpdCoresConnectedCoreId	M	N-Acc
docsRphyRpdCoresConnectedAddressType	M	RO
docsRphyRpdCoresConnectedAddress	M	RO
docsRphyRpdCoresConnectedIsPrincipal	M	RO
docsRphyRpdCoresConnectedName	M	RO
docsRphyRpdCoresConnectedVendorId	M	RO
docsRphyRpdCapabilitiesTable	M	N-Acc
docsRphyRpdCapabilitiesEntry	M	N-Acc
docsRphyRpdCapabNumBiDirPorts	M	RO
docsRphyRpdCapabNumDsPorts	M	RO
docsRphyRpdCapabNumUsPorts	M	RO
docsRphyRpdCapabNumTenGeNsPorts	M	RO
docsRphyRpdCapabNumOneGeNsPorts	M	RO
docsRphyRpdCapabNumDsScQamChans	M	RO
docsRphyRpdCapabNumDsOfdmChans	M	RO
docsRphyRpdCapabNumUsScQamChans	M	RO
docsRphyRpdCapabNumUsOfdmaChans	M	RO
docsRphyRpdCapabNumDsOob551Chans	M	RO
docsRphyRpdCapabNumUsOob551Chans	M	RO
docsRphyRpdCapabNumDsOob552Chans	M	RO
docsRphyRpdCapabNumUsOob552Chans	M	RO
docsRphyRpdCapabNumNdfChans	M	RO
docsRphyRpdCapabNumNdrChans	M	RO
docsRphyRpdCapabSupportsUdpEncap	M	RO
docsRphyRpdCapabNumDsPspFlowsPerChan	M	RO
docsRphyRpdCapabNumUsPspFlowsPerChan	M	RO
docsRphyRpdCapabNumAsynchVideoChans	M	RO

Object	CCAP	Access
docsRphyRpdCapabNumCwToneGens	M	RO
docsRphyRpdCapabLowestCwToneFreq	M	RO
docsRphyRpdCapabHighestCwToneFreq	M	RO
docsRphyRpdCapabMaxCwTonePwr	M	RO
docsRphyRpdCapabQamAsPilot	M	RO
docsRphyRpdChanReachabilityTable	M	N-Acc
docsRphyRpdChanReachabilityEntry	M	N-Acc
docsRphyRpdChanReachabilityEnetPortIndex	M	N-Acc
docsRphyRpdChanReachabilityRfPortIndex	M	N-Acc
docsRphyRpdChanReachabilityChanType	M	N-Acc
docsRphyRpdChanReachabilityStartChanIndex	M	RO
docsRphyRpdChanReachabilityEndChanIndex	M	RO
docsRphyRpdDsUsRfPortAllocTable	M	N-Acc
docsRphyRpdDsUsRfPortAllocEntry	M	N-Acc
docsRphyRpdDsUsRfPortAllocIndex	M	N-Acc
docsRphyRpdDsUsRfPortAllocDirection	M	N-Acc
docsRphyRpdDsUsRfPortAllocScQamChans	M	RO
docsRphyRpdDsUsRfPortAllocOfdmChans	M	RO
docsRphyRpdDsUsRfPortAllocOob551Chans	M	RO
docsRphyRpdDsUsRfPortAllocOob552Chans	M	RO
docsRphyRpdDsUsRfPortAllocNdChans	M	RO
docsRphyRpdL2tpSessionInfoTable	M	N-Acc
docsRphyRpdL2tpSessionInfoEntry	M	N-Acc
docsRphyRpdL2tpSessionInfoLocalLcceIpAddrType	M	N-Acc
docsRphyRpdL2tpSessionInfoLocalLcceIpAddr	M	N-Acc
docsRphyRpdL2tpSessionInfoLocalId	M	N-Acc
docsRphyRpdL2tpSessionInfoRemoteLcceIpAddrType	M	RO
docsRphyRpdL2tpSessionInfoRemoteLcceIpAddr	M	RO
docsRphyRpdL2tpSessionInfoRemoteId	M	RO
docsRphyRpdL2tpSessionInfoConnCtrlID	M	RO
docsRphyRpdL2tpSessionInfoUdpPort	M	RO
docsRphyRpdL2tpSessionInfoDescr	M	RO
docsRphyRpdL2tpSessionInfoSessionType	M	RO
docsRphyRpdL2tpSessionInfoSessionSubType	M	RO
docsRphyRpdL2tpSessionInfoMaxPayload	M	RO

Object	CCAP	Access
docsRphyRpdL2tpSessionInfoPathPayload	M	RO
docsRphyRpdL2tpSessionInfoRpdIfMtu	M	RO
docsRphyRpdL2tpSessionInfoErrorCode	M	RO
docsRphyRpdL2tpSessionInfoCreationTime	M	RO
docsRphyRpdL2tpSessionInfoOperStatus	M	RO
docsRphyRpdL2tpSessionInfoLocalStatus	M	RO
docsRphyRpdL2tpSessionInfoLastChange	M	RO
docsRphyRpdL2tpSessionStatsTable	M	N-Acc
docsRphyRpdL2tpSessionStatsEntry	M	N-Acc
docsRphyRpdL2tpSessionStatsOutOfSeqPkts	M	RO
docsRphyRpdPhysEntityTable	M	N-Acc
docsRphyRpdPhysEntityEntry	M	N-Acc
docsRphyRpdPhysEntityIndex	M	N-Acc
docsRphyRpdPhysEntityDescr	M	RO
docsRphyRpdPhysEntityVendorType	M	RO
docsRphyRpdPhysEntityContainedIn	M	RO
docsRphyRpdPhysEntityClass	M	RO
docsRphyRpdPhysEntityParentRelPos	M	RO
docsRphyRpdPhysEntityName	M	RO
docsRphyRpdPhysEntityHardwareRev	M	RO
docsRphyRpdPhysEntityFirmwareRev	M	RO
docsRphyRpdPhysEntitySoftwareRev	M	RO
docsRphyRpdPhysEntitySerialNum	M	RW
docsRphyRpdPhysEntityMfgName	M	RO
docsRphyRpdPhysEntityModelName	M	RO
docsRphyRpdPhysEntityAlias	M	RW
docsRphyRpdPhysEntityAssetID	M	RW
docsRphyRpdPhysEntityIsFRU	M	RO
docsRphyRpdPhysEntityMfgDate	M	RO
docsRphyRpdPhysEntityUris	M	RW
docsRphyRpdPhysEntityUUID	M	RO
docsRphyRpdPhysEntityIfIndex	M	RO
docsRphyRpdPhysEntSensorTable	M	N-Acc
docsRphyRpdPhysEntSensorEntry	M	N-Acc
docsRphyRpdPhysEntSensorType	M	RO

Object	CCAP	Access
docsRphyRpdPhysEntSensorScale	M	RO
docsRphyRpdPhysEntSensorPrecision	M	RO
docsRphyRpdPhysEntSensorValue	M	RO
docsRphyRpdPhysEntSensorOperStatus	M	RO
docsRphyRpdPhysEntSensorUnitsDisplay	M	RO
docsRphyRpdPhysEntSensorValueTimeStamp	M	RO
docsRphyRpdPhysEntSensorValueUpdateRate	M	RO
docsRphyRpdEnetIfTable	M	N-Acc
docsRphyRpdEnetIfEntry	M	N-Acc
docsRphyRpdEnetIfIndex	M	N-Acc
docsRphyRpdEnetIfDescr	M	RO
docsRphyRpdEnetIfName	M	RO
docsRphyRpdEnetIfAlias	M	RW
docsRphyRpdEnetIfType	M	RO
docsRphyRpdEnetIfMtu	M	RO
docsRphyRpdEnetIfPhysAddress	M	RO
docsRphyRpdEnetIfAdminStatus	M	RW
docsRphyRpdEnetIfOperStatus	M	RO
docsRphyRpdEnetIfLastChange	M	RO
docsRphyRpdEnetIfLinkUpDownTrapEnable	M	RW
docsRphyRpdEnetIfHighSpeed	M	RO
docsRphyRpdEnetIfPromiscuousMode	M	RW
docsRphyRpdEnetIfConnectorPresent	M	RO
docsRphyRpdEnetIfStatsTable	M	N-Acc
docsRphyRpdEnetIfStatsEntry	M	N-Acc
docsRphyRpdEnetIfStatsInOctets	M	RO
docsRphyRpdEnetIfStatsOutOctets	M	RO
docsRphyRpdEnetIfStatsInFrames	M	RO
docsRphyRpdEnetIfStatsOutFrames	M	RO
docsRphyRpdEnetIfStatsInUnicastOctets	M	RO
docsRphyRpdEnetIfStatsOutUnicastOctets	M	RO
docsRphyRpdEnetIfStatsInUnicastFrames	M	RO
docsRphyRpdEnetIfStatsOutUnicastFrames	M	RO
docsRphyRpdEnetIfStatsInMulticastOctets	M	RO
docsRphyRpdEnetIfStatsOutMulticastOctets	M	RO

Object	CCAP	Access
docsRphyRpdEnetIfStatsInMulticastFrames	M	RO
docsRphyRpdEnetIfStatsOutMulticastFrames	M	RO
docsRphyRpdEnetIfStatsInBroadcastOctets	M	RO
docsRphyRpdEnetIfStatsOutBroadcastOctets	M	RO
docsRphyRpdEnetIfStatsInBroadcastFrames	M	RO
docsRphyRpdEnetIfStatsOutBroadcastFrames	M	RO
docsRphyRpdEnetIfStatsInDiscards	M	RO
docsRphyRpdEnetIfStatsOutDiscards	M	RO
docsRphyRpdEnetIfStatsInErrors	M	RO
docsRphyRpdEnetIfStatsOutErrors	M	RO
docsRphyRpdEnetIfStatsInUnknownProtos	M	RO
docsRphyRpdEnetIfStatsCounterDiscontinuityTime	M	RO
docsRphyRpdIpv4GrpTable	M	N-Acc
docsRphyRpdIpv4GrpEntry	M	N-Acc
docsRphyRpdIpv4GrpIpForwarding	M	RW
docsRphyRpdIpv4GrpDefaultTTL	M	RW
docsRphyRpdIpv4GrpReasmTimeout	M	RO
docsRphyRpdIpv4GrpInterfaceTableLastChange	M	RO
docsRphyRpdIpv6GrpTable	M	N-Acc
docsRphyRpdIpv6GrpEntry	M	N-Acc
docsRphyRpdIpv6GrpIpForwarding	M	RW
docsRphyRpdIpv6GrpIpDefaultHopLimit	M	RW
docsRphyRpdIpv6GrpInterfaceTableLastChange	M	RO
docsRphyRpdIpv6GrpIfStatsTableLastChange	M	RO
docsRphyRpdIpv4InterfaceTable	M	N-Acc
docsRphyRpdIpv4InterfaceEntry	M	N-Acc
docsRphyRpdIpv4InterfaceIfIndex	M	N-Acc
docsRphyRpdIpv4InterfaceReasmMaxSize	M	RO
docsRphyRpdIpv4InterfaceEnableStatus	M	RW
docsRphyRpdIpv4InterfaceRetransmitTime	M	RO
docsRphyRpdIpv6InterfaceTable	M	N-Acc
docsRphyRpdIpv6InterfaceEntry	M	N-Acc
docsRphyRpdIpv6InterfaceIfIndex	M	N-Acc
docsRphyRpdIpv6InterfaceReasmMaxSize	M	RO
docsRphyRpdIpv6InterfaceIdentifier	M	RO

Object	CCAP	Access
docsRphyRpdIpv6InterfaceEnableStatus	M	RW
docsRphyRpdIpv6InterfaceReachableTime	M	RO
docsRphyRpdIpv6InterfaceRetransmitTime	M	RO
docsRphyRpdIpv6InterfaceForwarding	M	RW
docsRphyRpdIpIfStatsTable	M	N-Acc
docsRphyRpdIpIfStatsEntry	M	N-Acc
docsRphyRpdIpIfStatsIPVersion	M	N-Acc
docsRphyRpdIpIfStatsIfIndex	M	N-Acc
docsRphyRpdIpIfStatsInReceives	M	RO
docsRphyRpdIpIfStatsHCInReceives	M	RO
docsRphyRpdIpIfStatsInOctets	M	RO
docsRphyRpdIpIfStatsHCInOctets	M	RO
docsRphyRpdIpIfStatsInHdrErrors	M	RO
docsRphyRpdIpIfStatsInNoRoutes	M	RO
docsRphyRpdIpIfStatsInAddrErrors	M	RO
docsRphyRpdIpIfStatsInUnknownProtos	M	RO
docsRphyRpdIpIfStatsInTruncatedPkts	M	RO
docsRphyRpdIpIfStatsInForwDatagrams	M	RO
docsRphyRpdIpIfStatsHCInForwDatagrams	M	RO
docsRphyRpdIpIfStatsReasmReqds	M	RO
docsRphyRpdIpIfStatsReasmOKs	M	RO
docsRphyRpdIpIfStatsReasmFails	M	RO
docsRphyRpdIpIfStatsInDiscards	M	RO
docsRphyRpdIpIfStatsInDelivers	M	RO
docsRphyRpdIpIfStatsHCInDelivers	M	RO
docsRphyRpdIpIfStatsOutRequests	M	RO
docsRphyRpdIpIfStatsHCOutRequests	M	RO
docsRphyRpdIpIfStatsOutForwDatagrams	M	RO
docsRphyRpdIpIfStatsHCOutForwDatagrams	M	RO
docsRphyRpdIpIfStatsOutDiscards	M	RO
docsRphyRpdIpIfStatsOutFragReqds	M	RO
docsRphyRpdIpIfStatsOutFragOKs	M	RO
docsRphyRpdIpIfStatsOutFragFails	M	RO
docsRphyRpdIpIfStatsOutFragCreates	M	RO
docsRphyRpdIpIfStatsOutTransmits	M	RO

Object	CCAP	Access
docsRphyRpdIpIfStatsHCOutTransmits	M	RO
docsRphyRpdIpIfStatsOutOctets	M	RO
docsRphyRpdIpIfStatsHCOutOctets	M	RO
docsRphyRpdIpIfStatsInMcastPkts	M	RO
docsRphyRpdIpIfStatsHCInMcastPkts	M	RO
docsRphyRpdIpIfStatsInMcastOctets	M	RO
docsRphyRpdIpIfStatsHCInMcastOctets	M	RO
docsRphyRpdIpIfStatsOutMcastPkts	M	RO
docsRphyRpdIpIfStatsHCOutMcastPkts	M	RO
docsRphyRpdIpIfStatsOutMcastOctets	M	RO
docsRphyRpdIpIfStatsHCOutMcastOctets	M	RO
docsRphyRpdIpIfStatsInBcastPkts	M	RO
docsRphyRpdIpIfStatsHCInBcastPkts	M	RO
docsRphyRpdIpIfStatsOutBcastPkts	M	RO
docsRphyRpdIpIfStatsHCOutBcastPkts	M	RO
docsRphyRpdIpIfStatsDiscontinuityTime	M	RO
docsRphyRpdIpIfStatsRefreshRate	M	RO
docsRphyRpdIpAddressTable	M	N-Acc
docsRphyRpdIpAddressEntry	M	N-Acc
docsRphyRpdIpAddressAddrType	M	N-Acc
docsRphyRpdIpAddressAddr	M	N-Acc
docsRphyRpdIpAddressIfIndex	M	RC
docsRphyRpdIpAddressType	M	RC
docsRphyRpdIpAddressPrefixLen	M	RO
docsRphyRpdIpAddressOrigin	M	RO
docsRphyRpdIpAddressStatus	M	RC
docsRphyRpdIpAddressCreated	M	RO
docsRphyRpdIpAddressLastChanged	M	RO
docsRphyRpdIpNetToPhysicalTable	M	N-Acc
docsRphyRpdIpNetToPhysicalEntry	M	N-Acc
docsRphyRpdIpNetToPhysicalIfIndex	M	N-Acc
docsRphyRpdIpNetToPhysicalNetAddressType	M	N-Acc
docsRphyRpdIpNetToPhysicalNetAddress	M	N-Acc
docsRphyRpdIpNetToPhysicalPhysAddress	M	RC
docsRphyRpdIpNetToPhysicalLastUpdated	M	RO

Object	CCAP	Access
docsRphyRpdIpNetToPhysicalType	M	RC
docsRphyRpdIpNetToPhysicalState	M	RO
docsRphyRpdIpDefaultRouterTable	M	N-Acc
docsRphyRpdIpDefaultRouterEntry	M	N-Acc
docsRphyRpdIpDefaultRouterAddressType	M	N-Acc
docsRphyRpdIpDefaultRouterAddress	M	N-Acc
docsRphyRpdIpDefaultRouterIfIndex	M	N-Acc
docsRphyRpdIpDefaultRouterLifetime	M	RO
docsRphyRpdIpDefaultRouterPreference	M	RO
docsRphyRpdIpIcmpStatsTable	M	N-Acc
docsRphyRpdIpIcmpStatsEntry	M	N-Acc
docsRphyRpdIpIcmpStatsIPVersion	M	N-Acc
docsRphyRpdIpIcmpStatsInMsgs	M	RO
docsRphyRpdIpIcmpStatsInErrors	M	RO
docsRphyRpdIpIcmpStatsOutMsgs	M	RO
docsRphyRpdIpIcmpStatsOutErrors	M	RO
docsRphyRpdIpIcmpMsgStatsTable	M	N-Acc
docsRphyRpdIpIcmpMsgStatsEntry	M	N-Acc
docsRphyRpdIpIcmpMsgStatsIPVersion	M	N-Acc
docsRphyRpdIpIcmpMsgStatsType	M	N-Acc
docsRphyRpdIpIcmpMsgStatsInPkts	M	RO
docsRphyRpdIpIcmpMsgStatsOutPkts	M	RO
docsRphyCcapL2tpSessionInfoTable	M	N-Acc
docsRphyCcapL2tpSessionInfoEntry	M	N-Acc
docsRphyCcapL2tpSessionInfoLocalLcceIpAddrType	M	N-Acc
docsRphyCcapL2tpSessionInfoLocalLcceIpAddr	M	N-Acc
docsRphyCcapL2tpSessionInfoLocalId	M	N-Acc
docsRphyCcapL2tpSessionInfoRemoteLcceIpAddrType	M	RO
docsRphyCcapL2tpSessionInfoRemoteLcceIpAddr	M	RO
docsRphyCcapL2tpSessionInfoRemoteId	M	RO
docsRphyCcapL2tpSessionInfoCoreId	M	RO
docsRphyCcapL2tpSessionInfoConnCtrlID	M	RO
docsRphyCcapL2tpSessionInfoUdpPort	M	RO
docsRphyCcapL2tpSessionInfoDescr	M	RO
docsRphyCcapL2tpSessionInfoSessionType	M	RO

Object	CCAP	Access
docsRphyCcapL2tpSessionInfoSessionSubType	M	RO
docsRphyCcapL2tpSessionInfoMaxPayload	M	RO
docsRphyCcapL2tpSessionInfoPathPayload	M	RO
docsRphyCcapL2tpSessionInfoCoreIfMtu	M	RO
docsRphyCcapL2tpSessionInfoIncludeDOCSISMsgs	M	RO
docsRphyCcapL2tpSessionInfoErrorCode	M	RO
docsRphyCcapL2tpSessionInfoCreationTime	M	RO
docsRphyCcapL2tpSessionInfoOperStatus	M	RO
docsRphyCcapL2tpSessionInfoLocalStatus	M	RO
docsRphyCcapL2tpSessionInfoLastChange	M	RO
docsRphyCcapL2tpSessionStatsTable	M	N-Acc
docsRphyCcapL2tpSessionStatsEntry	M	N-Acc
docsRphyCcapL2tpSessionStatsOutOfSeqPkts	M	RO
docsRphyCcapCinLatencyTable	M	N-Acc
docsRphyCcapCinLatencyEntry	M	N-Acc
docsRphyCcapCinLatencyLastVal	M	RO
docsRphyCcapCinLatencyLastValTime	M	RO
docsRphyCcapCinLatencyInterval	M	RO
docsRphyCcapCinLatencyPerfTable	M	N-Acc
docsRphyCcapCinLatencyPerfEntry	M	N-Acc
docsRphyCcapCinLatencyPerfIntervalSeq	M	RO
docsRphyCcapCinLatencyPerfVal	M	RO
docsRphyCcapCinLatencyPerfValTime	M	RO

Annex B Format and Content for Event, SYSLOG, and SNMP Notification (Normative)[46]

Table B-1 in this annex summarizes the format and content for event, syslog, and SNMP notifications required for DOCSIS 3.1-compliant CCAP Core.

Each row specifies a possible event that may appear in the CCAP Core. These events are to be reported by a cable device through local event logging, and may be accompanied by syslog or SNMP notification.

The "Process" and "Sub-Process" columns indicate in which stage the event happens. The "CCAP Priority" column indicates the priority the event is assigned in the CCAP Core. These priorities are the same as is reported in the docsDevEvLevel object in the cable device MIB [RFC 4639] and in the LEVEL field of the syslog.

The "Event Message" column specifies the event text, which is reported in the docsDevEvText object of the cable device MIB and the text field of the syslog. The "Message Notes And Details" column provides additional information about the event text in the "Event Message" column. Some of the text fields include variable information. The variables are explained in the "Message Notes And Details" column. For events where the "Event Message" or "Message Notes and Details" column includes either <P1> or <P2>, there is a single space between the value as defined by the <P1> or <P2> and the preceding text.

Example SNMP Notification and Syslog message "Event Message" text string for Event ID 69020900:

SNMP CVC Validation Failure SNMP Manager: 10.50.1.11;CM-MAC=00:22:ce:03:f4:da;CMTS-MAC=00:15:20:00:25:ab;CM-QOS=1.1;CM-VER=3.0;

This specification defines the following keywords as part of the "Event Message" column:

"<TAGS>" (without the quotes) corresponds to:

For the CMTS (without the quotes): ";<CM-MAC>;<CM-QOS>;<CM-VER>;<CMTS-VER>;"

Where:

<CM-MAC>: CM MAC Address;

Format*: "CM-MAC=xx:xx:xx:xx:xx:xx"

<CMTS-MAC>: CMTS MAC Address;

Format*: "CMTS-MAC=xx:xx:xx:xx:xx:xx"

<CM-QOS>: CM DOCSIS QOS Version;

Format*: "CM-QOS=1.0" or "CM-QOS=1.1"

<CM-VER>: CM DOCSIS Version;

Format*: "CM-VER=1.1" or "CM-VER=2.0" or "CM-VER=3.0" or "CM-VER=3.1"

<CMTS-VER>: CMTS DOCSIS Version;

Format*: "CMTS-VER=1.1" or "CMTS-VER=2.0" or "CMTS-VER=3.0" or "CMTS-VER=3.1"

For the RPD (without the quotes): ";<RPD-MAC>;<CCAP-IP>;<RPD-MHA-VER>;"

Where:

<RPD-MAC>: RPD MAC Address associated with the lowest numbered CIN facing Ethernet port;

Format*: "RPD-MAC=xx:xx:xx:xx:xx:xx "

[46] Modified by R-OSSI-N-16.1446-2 on 4/22/16 by KB.

```
<CCAP-IP>:   Primary CCAP Core IPv4 or IPv6 Address;

Format*: "CCAP-IP=nnn.nnn.nnn.nnn" or "CCAP-
IP=nnnn:nnnn:nnnn:nnnn:nnnn:nnnn:nnnn:nnnn"

<RPD-MHA-VER>:      RPD RPHY MHA Version;

Format*: "RPD-VER=2"
```

(*) without the quotes

The CCAP Core MUST support all events defined in Annex D of [CCAP-OSSIv3.1]. The CCAP Core MUST support all mandatory events as defined in Table B-1.

The RPD MUST support all events defined in Table B-2.

Example SNMP Notification and Syslog message "Event Message" text string for Event ID 69010100:

```
SW Download INIT - Via NMS SW file: junk.bin - SW server: 10.50.1.11;CM-
MAC=00:22:ce:03:f4:da;CMTS-MAC=00:15:20:00:25:ab;CM-QOS=1.1;CM-VER=3.0;
```

The CCAP Core and RPD MAY append additional vendor-specific text to the end of the event text reported in the docsDevEvText object and the syslog text field.

The "Error Code Set" column specifies the error code. The "Event ID" column indicates a unique identification number for the event, which is assigned to the docsDevEvId object in the cable device MIB and the <eventId> field of the syslog. The "Notification Name" column specifies the SNMP notification, which notifies this event to an SNMP notification receiver.

The syslog format, as well as the rules to uniquely generate an event ID from the error code, are described in Section 9.2.2.1.3 of this specification.

Table B-1 - CCAP Core Event Format and Content[47]

Process	Sub-Process	CCAP Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID	Notification Name
Authentication and Encryption							
RPD Init	IKE Mutual Authentication	Error	Mutual Authentication error; <P1>: <P2><TAGS>	P1 = RPD IPv4 or IPv6 Address P2 = {Certificate Revoked, Certificate Failed} P2 is a string selected from a list of pre-defined strings	B.700.0	66070000	docsDevCmtsEventNotif
Connectivity							
Connectivity	RPD	Error	RPD Connection Refused; <P1>: <P2><TAGS>	P1 = RPD IPv4 or IPv6 Address P2 = text string indicating reason	B.701.0	66070100	docsDevCmtsEventNotif
Connectivity	RPD	Warning	GCP Connection Closed – RPD Detected Multiple Active Principal Cores<TAGS>		B.701.1	66070101	docsDevCmtsEventNotif

Table B-2 - RPD Event Format and Content[48]

Process	Sub-Process	RPD Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
Authentication and Encryption						
Init	Network Authentication	Error	Network Authentication Error: <P1><TAGS>	P1 = Authentication error description	B.701.0	66070100
Init	Mutual Authentication	Error	Mutual Authentication Error: <P1><TAGS>	P1 should include a string based on a list of pre-defined strings P1 = {Certificate Revoked, Certificate Failed, Other}	B.701.1	66070101

[47] Modified per R-OSSI-N-16.1446-2 on 4/22/16 by KB.
[48] Modified per R-OSSI-N-16.1448-2 on 4/27/16 by KB.

Process	Sub-Process	RPD Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
Authentication	SSH	Notice	SSH Authentication Successful from: <P1><P2><TAGS>	P1 = IP Address of SSH client P2 = user id of SSH client	B.701.2	66070102
Authentication	SSH	Warning	SSH Authentication Error from:<P1><P2><TAGS>	P1 = IP Address of SSH client P2 = user id of SSH client	B.701.3	66070103
Connectivity						
Connectivity	CCAP Core	Error	Connection lost - Auxiliary CCAP Core<TAGS>		B.702.0	66070200
Connectivity	CCAP Core	Critical	Connection lost – Principal CCAP Core<TAGS>		B.702.1	66070101
Connectivity	CCAP Core	Error	Principal Core Not Found<TAGS>		B.702.2	66070102
Connectivity	CCAP Core	Warning	Multiple Active Principal Cores Found<TAGS>		B.702.3	66070103
Connectivity	GCP	Error	GCP Connection Failure<TAGS>		B.702.4	66070104
Connectivity	Synchronization	Error	Loss of Clock Sync<TAGS>		B.702.5	66070105
Connectivity	Synchronization	Notice	Clock Sync Reestablished<TAGS>		B.702.6	66070106
Connectivity	Synchronization	Warning	Loss of Clock Slave<TAGS>		B.702.7	66070107
Connectivity	Synchronization	Notice	Clock Slave Reestablished<TAGS>		B.702.8	66070108
Connectivity	L2TPv3	Error	L2TPv3 Connection Error<TAGS>		B.702.9	66070109
Connectivity	High Availability	Warning	Failover to Standby Core<TAGS>		B.702.10	66070110
Connectivity	High Availability	Warning	Failback to Active Core<TAGS>		B.702.11	66070111
Init	Reboot	Notice	Reboot <P1><P2><TAGS>	P1 = {warm start, cold start} P2 = Text string indicating reason	B.702.12	66070112
Connectivity	Ethernet Interface	Error	Ethernet Link Down:<P1><TAGS>	P1 = Ethernet port number	B.702.13	66070113

Process	Sub-Process	RPD Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
Connectivity	Ethernet Interface	Notice	Ethernet Link Up:<P1><TAGS>	P1 = Ethernet port number	B.702.14	66070114
Connectivity	Pseudowire Interface	Error	Pseudowire Connection Down:<P1><P2><TAGS>	P1 = RPD PW Session ID P2 = CCID	B.702.15	66070115
Connectivity	Pseudowire Interface	Notice	Pseudowire Connection Up:<P1><P2><TAGS>	P1 = RPD PW Session ID P2 = CCID	B.702.16	66070116
DHCP, TOD						
DHCP		Error	DHCP RENEW sent – No response for <P1><TAGS>	P1= IPv4 or IPv6	B.703.0	66070300
DHCP		Error	DHCP REBIND sent – No response for <P1><TAGS>	P1=IPv4 or IPv6	B.703.1	66070301
DHCP		Error	DHCP RENEW WARNING – Field invalid in response <P1> option<TAGS>	P1=v4	B.703.2	66070302
DHCP		Critical	DHCP RENEW FAILED - Critical field invalid in response<TAGS>		B.703.3	66070303
DHCP		Error	DHCP REBIND WARNING – Field invalid in response<TAGS>		B.703.4	66070304
DHCP		Critical	DHCP REBIND FAILED - Critical field invalid in response<TAGS>		B.703.5	66070305
DHCP		Notice	DHCP Reconfigure received<TAGS>		B.703.6	66070306
DHCP		Notice	DHCP Renew - lease parameters <P1> modified<TAGS>	P1 = list of params that changed at renew	B.703.7	66070307
DHCP		Error	Primary lease failed, IPv4 fallback initiated<TAGS>		B.703.8	66070308
DHCP		Critical	DHCP Failed - CCAP Core list missing<TAGS>		B.703.9	66070309
Init	DHCP	Critical	DHCP FAILED – Discover sent, no offer received<TAGS>		B.703.10	66070310
Init	DHCP	Critical	DHCP FAILED – Request sent, No response<TAGS>		B.703.11	66070311

Process	Sub-Process	RPD Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
Init	DHCP	Warning	DHCP WARNING - Non-critical field invalid in response <TAGS>		B.703.12	66070312
Init	DHCP	Critical	DHCP FAILED – Critical field invalid in response<TAGS>		B.703.13	66070313
Init	DHCP	Critical	DHCP failed – RS sent, no RA received<TAGS>		B.703.14	66070314
Init	DHCP	Critical	DHCP Failed – Invalid RA<TAGS>		B.703.15	66070315
Init	DHCP	Critical	DHCP failed – DHCP Solicit sent, No DHCP Advertise received<TAGS>		B.703.16	66070316
Init	DHCP	Critical	DHCP failed – DHCP Request sent, No DHCP REPLY received<TAGS>		B.703.17	66070317
Init	DHCP	Error	Primary address acquired, secondary failed<TAGS>		B.703.18	66070318
Init	DHCP	Error	Primary address failed, secondary active<TAGS>		B.703.19	66070319
Init	IPv6 Address Acquisition	Critical	Link-Local address failed DAD<TAGS>		B.703.20	66070320
Init	IPv6 Address Acquisition	Critical	DHCP lease address failed DAD<TAGS>		B.703.21	66070321
Init	TOD	Error	ToD request sent – No Response received<TAGS>		B.703.22	66070322
Init	TOD	Error	ToD Response received – Invalid data format<TAGS>		B.703.23	66070323
Secure Software Download						
SW Upgrade	SW UPGRADE INIT	Notice	SW Download INIT – Via RPD CLI	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.0	66070400

Process	Sub-Process	RPD Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
SW Upgrade	SW UPGRADE INIT	Notice	SW Download INIT – Via GCP	Other than Local Log, append: SW file: <P2> - SW server: < P3><TAGS> P2 = SW file name P3 = SW Download server IP address	B.704.1	66070401
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	SW Upgrade Failed during download – Max retry exceed (3)	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.2	66070402
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	SW Upgrade Failed Before Download – Server not Present	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.3	66070403
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	SW upgrade Failed before download – File not Present	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.4	66070404

Process	Sub-Process	RPD Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	SW upgrade Failed before download –TFTP Max Retry Exceeded	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.5	66070405
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	SW upgrade Failed after download – Incompatible SW file	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.6	66070406
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	SW upgrade Failed after download – SW File corruption	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.7	66070407
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	Disruption during SW download – Power Failure	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.8	66070408

Process	Sub-Process	RPD Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	Disruption during SW download – RF removed	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.9	66070409
SW Upgrade	SW UPGRADE SUCCESS	Notice	SW download Successful – Via RPD CLI	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.10	66070410
SW Upgrade	SW UPGRADE SUCCESS	Notice	SW download Successful – Via GCP	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.11	66070411
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	Improper Code File Controls	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.12	66070412

Process	Sub-Process	RPD Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	Code File Manufacturer CVC Validation Failure	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.13	66070413
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	Code File Manufacturer CVS Validation Failure	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.14	66070414
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	Code File Co-Signer CVC Validation Failure	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.15	66070415
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	Code File Co-Signer CVS Validation Failure	Other than Local Log, append: SW file: <P1> - SW server: < P2><TAGS> P1 = SW file name P2 = SW Download server IP address	B.704.16	66070416

Process	Sub-Process	RPD Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
SW Upgrade	VERIFICATION OF CVC	Error	Improper GCP CVC Format	Other than Local Log, append: Config file: <P1> - Config file server: <P2><TAGS> P1 = Config file name P2 = Config file server IP address	B.704.17	66070417
SW Upgrade	VERIFICATION OF CVC	Error	GCP CVC Validation Failure	Other than Local Log, append: Config file: <P1> - Config file server: <P2><TAGS> P1 = Config file name P2 = Config file server IP address	B.704.18	66070418
Physical and Environmental						
Environmental	Temperature	Warning	Sensor unit=<P1> - High Temperature Threshold Exceeded <P2><P3><TAGS>	P1 = entPhysicalIndex of temperature sensor P2 = Sensor reading (Celsius Temperature) P3 = value of sysUpTime of sensor reading	B.705.0	66070500
Environmental	Temperature	Warning	Sensor unit=<P1> - Normal Operating Temperature Exceeded: <P2><P3><TAGS>	P1 = entPhysicalIndex of temperature sensor P2 = Sensor reading (Celsius Temperature) P3 = value of sysUpTime of sensor reading	B.705.1	66070501

Process	Sub-Process	RPD Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
Physical	Power	Warning	Power supply unit=<P1> - Below 95%<P2><P3><TAGS>	P1 = entPhysicalIndex of powerSupply P2 = Sensor reading (Watts) P3 = value of sysUpTime of sensor reading	B.705.2	66070502
Physical	Power	Critical	Power Supply unit=<P1> - Improper Input Voltage<P2><P3><TAGS>	P1 = entPhysicalIndex of powerSupply P2 = Sensor reading (Volts) P3 = value of sysUpTime of sensor reading	B.705.3	66070503
Environmental	Security	Warning	Enclosure door opened <P1><TAGS>	P1 = value of sysUpTime of sensor reading	B.705.4	66070504
Environmental	Humidity	Warning	Sensor unit=<P1> - High Humidity Threshold Exceeded <P2><P3><TAGS>	P1 = entPhysicalIndex of humidity sensor P2 = Sensor reading (percent relative humidity) P3 = value of sysUpTime of sensor reading	B.705.5	66070505
Environmental	Humidity	Warning	Sensor unit=<P1> - Normal Operating Humidity Exceeded:<P2><P3><TAGS>	P1 = entPhysicalIndex of humidity sensor P2 = Sensor reading (percent relative humidity) P3 = value of sysUpTime of sensor reading	B.705.6	66070506
Environmental	Security	Warning	Local craft port accessed <TAGS>		B.705.7	66070507

B.1 Deprecated Events[49]

Table B-3 in this annex lists deprecated events, including any associated syslog and SNMP trap notifications for the events, for a DOCSIS 3.1-compliant CCAP Core or RPD. Implementation of deprecated events is optional.

Table B-3 - Deprecated Events

Process	Sub-Process	CCAP Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID	Notification Name
				No Deprecated Events are defined			

B.2 Example SNMP Notification and Syslog Event Message (Informative)

The following is an example SNMP Notification and Syslog message "Event Message" text string for Event ID 70000304:

```
Power - Power Supply Bus Failure; unit=pw/1/1/;
```

[49] Added per R-OSSI-N-16.1446-2 on 4/22/16 by KB.

Annex C Data Type Definitions (Normative)

C.1 Overview

This section includes the data type definitions for the Information Models defined for use in the CCAP. UML is used for modeling the management requirements.

The data types defined in this section are mapped for use with SNMP MIBs, IPDR XML schemas, YANG modules and XSD Schemas.

C.1.1 Data Types Mapping

XML is becoming the standard for data definition models. With XML, data transformations can be done with or without a model (DTD or Schema definition). DTDs and XML schemas provide additional data validation layer to the applications exchanging XML data. There are several models to map formal notation constructs like ASN.1 to XML [ITU-T X.692], UML to XML, YANG to XML, or XML by itself can be used for modeling purposes.

Each area of data information interest approaches XML and defines data models and/or data containment structures and data types. Similarly, SNMP took and modified a subset of ASN.1 for defining the Structured Management Information SMIv1 and SMIv2.

Due to the lack of a unified data model and data types for Network Management, a neutral model would be appropriate to allow capturing specific requirements and methodologies from existing protocols and allow forward or reverse engineering of those standards like SNMP to the general object model and vice versa.

C.1.2 Data Types Requirements and Classification

The Information Model has to provide seamless translation for SMIv2 requirements, in particular when creating MIB modules based on the Information Model. This specification needs to provide full support of [RFC 2578], [RFC 2579], and the clarifications and recommendations of [RFC 4181].

The Information Model has to provide seamless translation for YANG modeling requirements, in particular when creating YANG modules based on the Information Model.

Thus, there are two data type groups defined for modeling purposes and mapping to protocol data notation roundtrip.

- General data types

  Required data types to cover all the management syntax and semantic requirement for all OSSI supported data models. In this category are data types defined in SNMP SMIv2 [RFC 2578], and YANG common data types [RFC 6991].

- Extended data types

  Management protocols specialization based on frequent usage or special semantics. Required data types to cover all the syntax requirement for all OSSI supported data models. In this category are SNMP TEXTUAL-CONVENTION clauses [RFC 2579] of mandatory or recommended usage by [RFC 2579] and [RFC 4181] when modeling for SNMP MIB modules.

C.1.3 Data Type Mapping Methodology

The specification "XML Schema Part 2: Data types Second Edition" is based on [ISO 11404], which provides language-independent data types (see XML Schema reference). The mapping proposed below uses a subset of the XML schema data types to cover both SNMP forward and reverse engineering, and IPDR types. Any additional protocol being added should be feasible to provide the particular mappings.

SMIv2 has an extensive experience of data types for management purposes; for illustration consider Counter32 and Counter64 SMIv2 types [RFC 2578]. The XML schema data types makes no distinction of derived 'decimal' types and the semantics that are associated to counters, e.g., counters do not necessarily start at 0.

Most of the SNMP information associated to data types are reduced to size and range constraints and specialized enumerations.

C.1.4 General Data Types (SNMP Mapping)

Table represents the mapping between the OSSI object model General Types and their equivalent representation for SNMP MIB Modules and IPDR Service Definitions. The permitted values for the data types are indicated in terms of value ranges and string length when applicable. The OM Data Type column includes the data types to map to SNMP, using the appropriated type in the corresponding protocol if applicable or available. The SNMP Mapping references to SNMP data types are defined in [RFC 2578] or as described below.

Note that SNMP does not provide float, double or long XML-Schema data types. Also, SNMP might map a type to an SNMP subtyped value. For example, unsignedByte data type maps to Unsigned32 subtyped to the appropriate range indicated by the Permitted Values (0..255 in this case). Other data types are mapped to SNMP TEXTUAL-CONVENTIONS as indicated by the references.

Table C-1 - General Data Types[50]

OM Data Type	XML-Schema data type	Permitted Values	SNMP Mapping
Enum	int	-2147483648..2147483647	INTEGER
EnumBits	hexBinary		BITS
Int	int	-2147483648..2147483647	Integer32
unsignedInt	unsignedInt	0..4294967295	Unsigned32
long	long	-9223372036854775808..-9223372036854775807	N/A
unsignedLong	unsignedLong	0..18446744073709551615	CounterBasedGauge64 [RFC 2856]
hexBinary	hexBinary		OCTET STRING
string	string		SnmpAdminString [RFC 3411]
boolean	boolean		TruthValue [RFC 2579]
Byte	byte	-128..127	Integer32
unsignedByte	unsignedByte	0..255	Unsigned32
Short	short	-32768..32767	Integer32
unsignedShort	unsignedShort	0..65535	Unsigned32
Gauge32	unsignedInt		Gauge32
Counter32	unsignedInt		Counter32
Counter64	unsignedLong		Counter64
IpAddress	hexBinary	SIZE (4)	IpAddress
Opaque	hexBinary		Opaque
dateTime	dateTime		DateAndTime
dateTimeMsec	unsignedLong		CounterBasedGauge64 [RFC 2856]
InetAddressIPv4	hexBinary	SIZE (4)	InetAddressIPv4 [RFC 4001]
InetAddressIPv6	hexBinary	SIZE (16)	InetAddressIPv6 [RFC 4001]
InetAddress	hexBinary	SIZE (0..255)	InetAddress [RFC 4001]
InetAddressType	enumeration	unknown(0), ipv4(1), ipv6(2), ipv4z(3), ipv6z(4), dns(16)	InetAddressType [RFC 4001]
Uuid	hexBinary		OCTET STRING
MacAddress	hexBinary	SIZE (6)	MacAddress

[50] Modified per R-OSSI-N-16.1469-1 on 4/27/16 by KB.

C.1.5 Primitive Data Types (YANG Mapping)

Table represents the mapping between the CCAP primitive data types and their equivalent representation in YANG. The permitted values for the data types are indicated in terms of value ranges and string length when applicable. The UML Primitive Data Type column includes the data types to map to YANG, using the appropriate type in YANG. The YANG Built-In Data Type Mapping references YANG data types defined in [RFC 6021] or as described below.

Table C-2 - Primitive Data Types

UML Primitive Data Type	YANG Data Type Mapping	Permitted Values
HexBinary	ccap-octet-data-type	([0-9a-fA-F]{2})*
EnumBits	bits	
Boolean	boolean	true, false
Enum	enumeration	-2147483648..2147483647
Byte	int8	-128..127
Short	int16	-32768..32767
Integer	int32	-2147483648..2147483647
Long	int64	-9223372036854775808..9223372036854775807
String	string	
UnsignedByte	uint8	0..255
UnsignedShort	uint16	0..65535
UnsignedInt	uint32	0..4294967295
UnsignedLong	uint64	0..18446744073709551615

C.1.6 Extended Data Types (SNMP Mapping)

There are two sources of Extended Data Types: Protocol specific data types, and OSSI data types.

SNMP derived types are defined in SNMP MIB Modules. The most important are in [RFC 2579], which is part of SNMP STD 58, and are considered in many aspects part of the SNMP protocol. Other MIB modules TEXTUAL-CONVENTION definitions have been adopted and recommended (e.g., [RFC 4181]) for re-usability and semantics considerations in order to unify management concepts; some relevant RFCs that include commonly used textual conventions are [RFC 4001], [RFC 2863], [RFC 3411], and [RFC 3419] among others (see [RFC 4181]).

Table includes the most relevant data types taken from SNMP to provide a direct mapping of the OSSI object model to SNMP MIB modules. For example, TagList comes from [RFC 3413] SnmpTaglist and preserves its semantics; AdminString comes from [RFC 3411] SnmpAdminString.

In general, when an OSSI object model needs to reference an existing SNMP textual convention for the purpose of round trip design from UML to SNMP, these textual conventions can be added to this list. Other sources of textual conventions not listed here are from MIB modules specific to DOCSIS, either as RFCs or Annex documents in this specification. Some of those sources are [RFC 4546] and Annex A.

OSSI data types are also defined in this specification in the Data Type section of OSSI annexes; for example, Annex A.

Table C-3 - Extended Data Types

OM Data Type	XML-Schema data type	Permitted Values	SNMP Mapping
PhysicalIndexOrZero	unsignedInt	0..2147483647	Integer32
TagList	string	SIZE (0..255)	SnmpTaglist
AdminString	string	SIZE (0..255)	SnmpAdminString
RowStatus	int		RowStatus
TimeStamp	unsignedInt		TimeStamp
duration	unsignedInt	0..2147483647	TimeInterval

OM Data Type	XML-Schema data type	Permitted Values	SNMP Mapping
StorageType	int		StorageType
InetAddressPrefixLength	unsignedInt	0..2040	Unsigned32
InetPortNumber	unsignedInt	0..65535	Unsigned32
DocsisQosVersion	int		DocsisQosVersion [RFC 4546]
DocsisUpstreamType	int		DocsisUpstreamType [RFC 4546]
DocsEqualizerData	hexBinary		DocsEqualizerData [RFC 4546]
TenthdBmV	int		TenthdBmV [RFC 4546]
TenthdB	int		TenthdB [RFC 4546]

C.1.7 Derived Data Types (YANG Mapping)

Table represents the mapping between the CCAP derived data types and their equivalent representation in YANG. The permitted values for the data types are indicated in terms of value ranges and string length when applicable. The UML Derived Data Type column includes the data types to map to YANG, using the appropriate type in YANG. The YANG Derived Data Type Mapping references YANG data types defined in [RFC 6021] or as described below.

Table C-4 - Derived Data Types

UML Derived Data Type	YANG Derived Data Type Mapping	Permitted Values
Counter32	counter32	
Counter64	counter64	
Gauge32	gauge32	
TimeStamp	timestamp	
MacAddress	mac-address	e.g., 01:23:45:67:89:ab
InetPortNumber	port-number	0..65535
IPAddress	ip-address	IPv4 or IPv6 Address
IPv4Address	ipv4-address	IPv4 Address
IPv6Address	ipv6-address	IPv6 Address
InetAddressPrefixLength	address-prefix-len-type	0..2040
InetIpv4Prefix	ipv4-prefix	IPv4 Address "/" IPv4 Prefix Length
InetIpv6Prefix	ipv6-prefix	IPv6 Address "/" IPv6 Prefix Length
Uri	uri	
TagList	snmp-tag-list-type	String(SIZE(0..255))
AdminState	admin-state-type	other(1), up(2), down(3), testing(4)
DateTime	date-and-time	

C.2 Remote PHY Common Data Type Definitions[51]

There are no additional data types created specifically to support the Remote PHY Information Models.

Refer to the CCAP Data Type Definitions section of [CCAP-OSSIv3.1] for common data type definitions that may be used in the Remote PHY Information Models.

[51] Modified per R-OSSI-N-16.1469-1 on 4/27/16 by KB.

Appendix I Information Modeling for OSSI (Informative)[52]

I.1 Information Model Notation

The Unified Modeling Language (UML) is a unified model for object-oriented analysis and design (OOA&D). UML is an OMG standard and is an accepted ISO specification [ISO 19501].

UML defines a general-purpose, graphical modeling language that can be applied to any application domain (e.g., communications) and implementation platforms (e.g., J2EE).

The OSSI Information Model diagram is represented by the UML Class Diagram. The class diagram describes the types of objects existing in a system and their static relationship or association.

I.1.1 Classes

Classes are generally represented by a square box with three compartments. The top compartment contains the class name (used here as the object name) with the first letter capitalized. The middle compartment contains the list of attributes with the first letter of each attribute in lower case. The bottom compartment contains the list of operations. For the purposes of this specification, the methods section of the class box is not used (suppressed) and the implementation level details of the attributes are omitted.

Attributes also include a visibility notation which precedes the attribute name and is one of the following:

- '+' public (default)
- '-' private
- '#' protected

If the above notation is omitted from the attribute, the default of public is implied. For the purposes of this specification, the protected visibility generally refers to indexes of MIB tables, schema instances, etc.

An interface is represented in the class diagram as an object with the keyword <<interface>> preceding the object name. In general, an interface is a declaration of a set of public features and obligations (such as get methods).

I.1.2 Associations

A class diagram also contains associations which represent relationships between instances of classes. An association has two ends with each end attached to one of the classes. The association end also has a multiplicity indicator which defines how many objects may participate in the relationship. Multiplicity notation is as follows:

- '1' exactly one
- '*' zero or more (default)
- '0..1' zero or one (optional)
- 'm..n' numerically specified

If the above notation is omitted from the association end, the default of '*' is implied.

If one end of the association contains an open arrowhead, this implies navigability in the direction indicated by the arrow.

I.1.3 Generalization

Generalization is the concept of creating subclasses from superclasses and is also known as inheritance within programming languages. Subclasses include (or inherit) all the elements of the superclass and may override inherited methods. Subclasses are more specific classes while superclasses are generalized classes.

The UML notation for Generalization is shown as a line with a hollow triangle as an arrowhead pointing to the generalized class.

[52] Moved from Section to Appendix by R-OSSI-N-16.1436-1 on 4/22/16 by KB.

I.1.4 Dependencies

Dependencies between two classes are represented by a dashed arrow between two objects. The object at the tail of the arrow depends on the object at the other end.

I.1.5 Comment

A Comment in a class diagram is a textual annotation attached to any element. This is represented as a note symbol with a dashed line connecting the note with the element.

I.1.6 Diagram Notation

Figure I-1 highlights the UML Class Diagram notation discussed in this section. Figure I-1 is not a complete representation of the UML Class Diagram notation, but captures those concepts used throughout this specification.

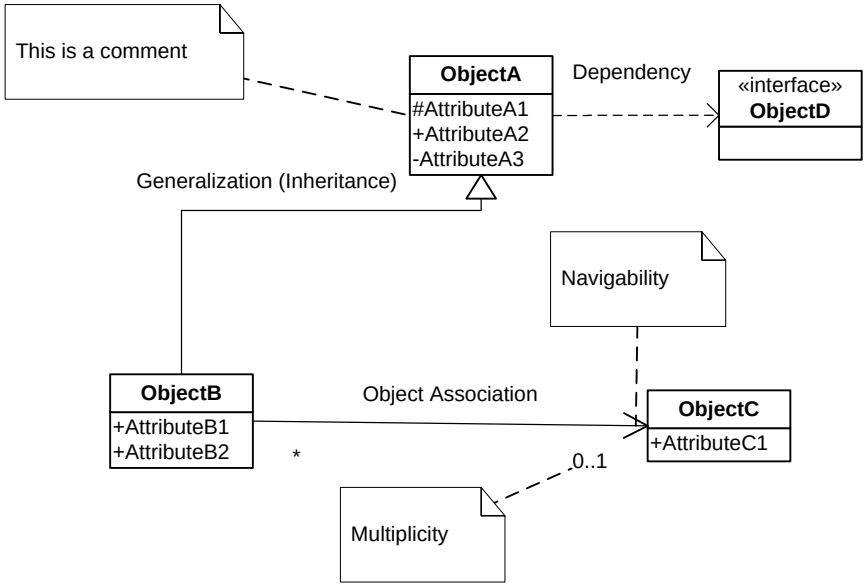

Figure I-1 - Object Model UML Class Diagram Notation

I.2 Object Instance Diagram

An Object Instance Diagram represents the objects in a system during one snapshot in time. In this diagram, the class objects are instantiated.

Figure I-2 shows an Object Instance Diagram for an instantiation (myObjectA) of ObjectA from Figure I-1.

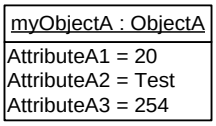

Figure I-2 - Object Instance Diagram for ObjectA

I.3 ObjectA Definition Example

This section defines the details of the object and its associated attributes as defined in the object model diagram. The description of the object includes behavior, persistence requirements (if any), object creation and deletion behavior (if any), etc.

Table I–1 lists the attributes the object defines in the object model. The object table is derived from the object model diagram where each row in the table represents an attribute of the object.

The "Attribute Name" column contains each defined attribute of the object. The naming convention for attributes is to capitalize the first letter and each letter of successive words within the name. Also, attribute names typically do not include any of the object name elements since this would cause duplication when the object and attributes are realized in SNMP.

The "Type" column contains the data type for the attribute. The data type can be a simple type such as unsignedInt or a defined data type such as EnumBits. DOCSIS 3.0 data types are defined in Annex C.

The "Access" column indicates the attributes accessibility (as mapped to an SNMP object for example). Example values include "key", "read-only", "read-write", and "read-create".

The "Type Constraints" column lists constraints on the normal data type specified in the "Type" column. If there are no defined constraints for the attribute, this column is empty. The example below for AttributeA1 lists a constraint on the unsignedInt Type where the range starts from 1 instead of normally starting from 0 for an unsignedInt.

The "Units" column lists units for the attribute or "N/A" if the attribute does not have units.

The "Default" column contains the default value for the attribute or "N/A" if the attribute does not have a default value or in cases where the attribute's description defines rules for the initialization value.

The sections following Table I–1 are attribute descriptions which might include behavioral requirements or references.

Table I–1 - ObjectA Example Table Layout

Attribute Name	Type	Access	Type Constraints	Units	Default
AttributeA1	unsignedInt	key	1..4294967295	N/A	N/A
AttributeA2	AdminString	read-write	SIZE (1..15)	N/A	N/A
AttributeA3	unsignedByte	read-create		seconds	60

I.3.1 AttributeA1

AttributeA1 is a key defined for…

NOTE: Objects which represent a table (in an SNMP MIB realization) and have N number of instances need to include at least one "key" attribute which is used to denote the instance or id. Key attributes are typically denoted with a protected visibility whereas all other attributes are denoted with a public visibility.

I.3.2 AttributeA2

AttributeA2 is …

NOTE: Persistence requirements are documented at the object level, not at the attribute level.

I.3.3 AttributeA3

AttributeA3 is …

I.4 Common Terms Shortened

The following table lists common terms which have been shortened to allow shorter SNMP MIB names. These shortened names are desired to be used consistently throughout the object models, SNMP MIBs and IPDR schemas. However, in some cases it might not be possible to maintain parity with pre-3.0 DOCSIS requirements.

Table I–2 - Shortened Common Terms

Original Word	Shortened Word
Address	Addr
Aggregate	Agg

Original Word	Shortened Word
Algorithm	Alg
Application	App
Attribute	Attr
Authorization	Auth
Channel	Ch
Command	Cmd
Config*	Cfg
Control	Ctrl
Default	Def
Destination	Dest
Direction	Dir
Downstream	Ds
Encryption	Encrypt
Equalization	Eq
Group	Grp
Length	Len
Maximum	Max
Minimum	Min
Multicast	Mcast
Provision*	Prov
Receive	Rx
Registration	Reg
Replication	Repl
Request	Req
Resequence	Reseq
Resequencing	Reseq
Response	Rsp
Segment	Sgmt
Sequence	Seq
Service	Svc
ServiceFlow	Sf
Session(s)	Sess
Source	Src
Threshold	Thrshld
Total	Tot
Transmit	Tx
Upstream	Us
* indicates a wildcard	

I.4.1 Exceptions

Data types and managed objects do not consistently use the shortened names. Also, the term ServiceFlowId remains unchanged. Service and ServiceFlow are often not shortened to retain backward compatibility with QoS managed objects.

Appendix II Sample CCAP XML Configuration (Informative)

II.1 CCAP XML Configuration File

To be provided in a future version of this specification.

Appendix III Acknowledgments (Informative)

On behalf of the cable industry and our member companies, CableLabs would like to thank the following individuals for their contributions to the development of this specification.

Contributor	Company Affiliation
Tom Ferreira	Arris
Niki Pantelias	Broadcom
Nikhil Tayal	CableLabs
Pawel Sowinski	Cisco
Joe Solomon	Comcast
John Bevilacqua	Comcast
Andrew Sundelin	Dial in the Sun, LLC
Michael Patrick	Harmonic, Inc.
Rei Brockket	Pace

Appendix IV Revision History

IV.1 Engineering Changes incorporated into CM-SP-R-OSSI-I02-160121

ECN Identifier	Accepted Date	Title of EC	Author
R-OSSI-N-15.1378-1	10/14/2015	New control objects for R-PHY	Solomon
R-OSSI-N-15.1407-1	12/16/2015	Section 11 SNMP MIB requirements and Annex A updates	Tayal
R-OSSI-N-15.1411-3	12/16/2015	Section 8.1 Secure Shell Requirements	Ferreira
R-OSSI-N-15.1412-4	12/16/2015	Section 11 SNMP MIB requirements, R-PHY MIB, and Annex A updates	Solomon

IV.2 Engineering Changes incorporated into CM-SP-R-OSSI-I03-160512

ECN Identifier	Accepted Date	Title of EC	Author
R-OSSI-N-15.1409-2	1/27/2016	Updates to RPD Secure Software Download for R-OSSI	Sowinski
R-OSSI-N-16.1429-1	3/2/2016	ifType and Annex A N-ACC corrections	Hedstrom
R-OSSI-N-16.1436-1	3/10/2016	Move section 6 to Appendix/Annex	Hedstrom
R-OSSI-N-16.1442-1	3/24/2016	Move section 11.3.7 info model to section 8	Hedstrom
R-OSSI-N-16.1443-2	3/24/2016	Updates to Session Reporting	Solomon
R-OSSI-N-16.1446-2	3/24/2016	Event Definitions for CCAP Core	Hedstrom
R-OSSI-N-16.1448-2	4/14/2016	Event Definitions and Mechanism for RPD	Hedstrom
R-OSSI-N-16.1454-1	4/14/2016	Align CCAP Core Configuration of CW Tones with other specifications	Sowinski
R-OSSI-N-16.1461-3	4/21/2016	Addition of DOCS-RPHY-MIB	Hedstrom
R-OSSI-N-16.1465-2	4/21/2016	R-PHY security certificates	Hedstrom
R-OSSI-N-16.1466-1	4/21/2016	RPHY OSSI YANG and XSD Modules	Hedstrom
R-OSSI-N-16.1469-1	4/21/2016	Clarification of RPD Control Address	Hedstrom
R-OSSI-N-16.1475-2	4/21/2016	Correction from previous ECNs and updates to section 10	Kim
R-OSSI-N-16.1494-1	4/21/2016	Move SSH requirements to Section 12	Hedstrom