SCTE | STANDARDS

Interface Practices Subcommittee

AMERICAN NATIONAL STANDARD

ANSI/SCTE 283 2023

Information Model for Smart Broadband Amplifiers

NOTICE

The Society of Cable Telecommunications Engineers (SCTE) Standards and Operational Practices (hereafter called "documents") are intended to serve the public interest by providing specifications, test methods and procedures that promote uniformity of product, interoperability, interchangeability, best practices, and the long term reliability of broadband communications facilities. These documents shall not in any way preclude any member or non-member of SCTE from manufacturing or selling products not conforming to such documents, nor shall the existence of such standards preclude their voluntary use by those other than SCTE members.

SCTE assumes no obligations or liability whatsoever to any party who may adopt the documents. Such adopting party assumes all risks associated with adoption of these documents and accepts full responsibility for any damage and/or claims arising from the adoption of such documents.

NOTE: The user's attention is called to the possibility that compliance with this document may require the use of an invention covered by patent rights. By publication of this document, no position is taken with respect to the validity of any such claim(s) or of any patent rights in connection therewith. If a patent holder has filed a statement of willingness to grant a license under these rights on reasonable and nondiscriminatory terms and conditions to applicants desiring to obtain such a license, then details may be obtained from the standards developer. SCTE shall not be responsible for identifying patents for which a license may be required or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Patent holders who believe that they hold patents which are essential to the implementation of this document have been requested to provide information about those patents and any related licensing terms and conditions. Any such declarations made before or after publication of this document are available on the SCTE web site at https://scte.org.

140 Philips Road
Exton, PA 19341

Document Types and Tags

Document Tags:

☐ Test or Measurement
☐ Architecture or Framework
☐ Procedure, Process or Method
☐ Checklist
☐ Metric
☐ Cloud
☐ Facility
☒ Access Network
☐ Customer Premises

Document Release History

Release	Date
SCTE 283 2023	*6/12/2023*

Note: Standards that are released multiple times in the same year use: a, b, c, etc. to indicate normative balloted updates and/or r1, r2, r3, etc. to indicate editorial changes to a released document after the year.

Table of Contents

Title **Page Number**

List of Figures

Title **Page Number**

List of Tables

Title **Page Number**

1. Introduction

1.1. Executive Summary

This standard provides an information model to be used for all amplifier communications in broadband communications networks. The model can be used for FDD and for FDX amplifiers and is applicable to stand-alone distribution amplifiers and to launch amplifiers inside nodes.

1.2. Scope

This standard defines an information model for communications with amplifiers used in hybrid fiber-coax (HFC) networks. The information model includes capabilities, configuration and status information which can be set either over a coaxial cable transponder or locally via direct wired or wireless connection. This release of the standard is compatible with [SCTE 279] amplifiers and could also be applicable to stand-alone FDX amplifiers and to launch amplifiers inside nodes.

The scope of this document is the amplifier and does not include consideration of the coaxial cable transponder.

1.3. Benefits

This standard defines an information model for broadband amplifiers, especially those used in DOCSIS 4.0 networks. The cable industry will benefit from having a standardized single information model that is used for all amplifier communications.

1.4. Intended Audience

This standard applies to all hybrid fiber-coax system operators and all node and amplifier equipment vendors.

1.5. Areas for Further Investigation or to be Added in Future Versions

This document reflects [SCTE 279] and if that document is updated, this document may also need updates.

Further updates could include items specific to fiber node launch amplifiers and/or FDX amplifiers.

Reporting capabilities such as RF test points and RF spectrum analysis are described in this document. In the future additional reporting capabilities may be deemed useful and be added to the information model.

2. Normative References

The following documents contain provisions which, through reference in this text, constitute provisions of this document. The editions indicated were valid at the time of subcommittee approval. All documents are subject to revision and, while parties to any agreement based on this document are encouraged to investigate the possibility of applying the most recent editions of the documents listed below, they are reminded that newer editions of those documents might not be compatible with the referenced version.

2.1. SCTE References

[SCTE 279] SCTE 279 2022, 1.8 GHz Broadband Radio Frequency Hardline Amplifiers for Cable Systems.

2.2. Standards from Other Organizations

[CANN] CableLabs Assigned Names and Numbers, CL-SP-CANN-I21-220831, August 31, 2022.

[CCAP-YANG] CCAP YANG Model for Event Messaging, CCAPevents.yang, http://www.cablelabs.com/YANG/DOCSIS

[CCAP-OSSIv4.0] CableLabs Data-Over-Cable Service Interface Specifications DOCSIS® 4.0 CCAP™ Operations Support System Interface Specification, CM-SP-CCAP-OSSIv4.0-I08-220629, November 16, 2022.

[CCAP-OSSIv3.1] CableLabs Data-Over-Cable Service Interface Specifications DOCSIS® 3.1 CCAP™ Operations Support System Interface Specification, CM-SP-CCAP-OSSIv3.1-I24-220518, May 18, 2022.

[CM-OSSIv4.0] CableLabs Data-Over-Cable Service Interface Specifications DOCSIS 4.0 Cable Modem Operations Support System Interface Specification, CM-SP-CM-OSSIv4.0-I07-221116, November 16, 2022.

[eDOCSIS] CableLabs Data-Over-Cable Service Interface Specifications Embedded DOCSIS Specification, CM-SP-eDOCSIS-I31-220831, August 31, 2022.

[FMA-OSSI] CableLabs Data-Over-Cable Service Interface Specifications FMA OSS Interface Specification, CM-SP-FMA-OSSI-I02-220602, June 2, 2022.

[IEEE 754] IEEE STD 754-2019, IEEE Standard for Floating-Point Arithmetic, IEEE Computer Society, July 7, 2019.

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

[ISO 6709] ISO 6709:2008, Standard representation of geographic point location by coordinates.

[PNM-3.1] CableLabs Data-Over-Cable Service Interface Specification Proactive Network Maintenance, CM-GL-PNM-3.1-V03-220118], January 18, 2022.

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

[RFC 3014] IETF RFC 3014, Notification Log MIB, November 2000.

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

[RFC 3411] IETF RFC 3411, 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 4293] IETF RFC 4293, Management Information Base for the Internet Protocol (IP), April 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 6020] IETF RFC 6020, YANG – A Data Modeling Language for the Network Configuration Protocol (NETCONF), October 2010.

[RFC 8341] IETF RFC 8341, Network Configuration Access Control Model, March 2018.

[RPHY-OSSI] CableLabs Data-Over-Cable Service Interface Specifications, Remote PHY OSS Interface Specification, CM-SP-R-OSSI-I18-220613, June 13, 2022.

2.3. Other Published Materials

No normative references are applicable.

3. Informative References

The following documents might provide valuable information to the reader but are not required when complying with this document.

3.1. SCTE References

No informative references are applicable.

3.2. Standards from Other Organizations

[UML Guidelines] UML Modeling Guidelines, CM-GL-OSS-UML-V01-180627, June 27, 2018, Cable Television Laboratories, Inc.

[ZTA-NIST] Zero Trust Architecture, National Institute of Standards and Technology (NIST), NIST Special Publication 800-207, August 2020.

[ZTA-EU] Technical Guidelines for the implementation of minimum security measures for Digital Service Providers, European Union Agency For Network And Information Security, December 2016.

3.3. Other Published Materials

No informative references are applicable.

4. Compliance Notation

shall	This word or the adjective "***required***" means that the item is an absolute requirement of this document.
shall not	This phrase means that the item is an absolute prohibition of this document.
forbidden	This word means the value specified ***shall*** never be used.
should	This word or the adjective "*recommended*" 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 or the adjective "*optional*" indicate a course of action permissible within the limits of the document.
deprecated	Use is permissible for legacy purposes only. Deprecated features *may* be removed from future versions of this document. Implementations *should* avoid use of deprecated features.

5. Abbreviations and Definitions

5.1. Abbreviations

AGC	automatic gain control
AT	attenuation stage
DAA	distributed access architecture
DHCP	dynamic host configuration protocol
DNS	domain name server
DOCSIS	Data Over Cable Service Interface Specifications
eAMP	embedded amplifier logical function
eCM	embedded cable modem logical function
EQ	equalization stage
FDD	frequency division duplex
FDX	full duplex
FQDN	Fully qualified domain name
gNMI	gRPC network management interface
GPS	global positioning system
gRPC	gRPC remote procedure call
HFC	hybrid fiber-coax
HTTPS	hypertext transfer protocol secure
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet protocol
JTAG	Joint Test Action Group
OSSI	operations support system interface
OUI	organizationally unique identifier
PKI	public key infrastructure
PNM	proactive network maintenance

RF	radio frequency
RMS	root mean square
RPC	remote procedure call
ro	read-only
rw	read-write
SCTE	Society of Cable Telecommunications Engineers
SPI	serial peripheral interface
TLS	transport layer security
TFTP	trivial file transfer protocol
UML	unified modeling language
USB	universal serial bus
YANG	yet another next generation
ZTA	zero trust architecture

5.2. Definitions

Definitions of terms used in this document are provided in this section. Defined terms that have specific meanings are capitalized. When the capitalized term is used in this document, the term has the specific meaning as defined in this section.

Downstream	The direction of signal transmission from the headend, hub site, or optical node to the subscriber. Also called forward.
Upstream	The direction of signal transmission from the subscriber to the headend, hub site, or node. Also called return or reverse.
Frequency division duplex	This describes a bi-directional broadband RF communications system where the downstream and upstream each have their own dedicated non-overlapping frequency spectrums.
Network	A network is a group of interconnected devices that can exchange messages using a variety of technologies.

6. Amplifier Management Overview

6.1. Relation to other SCTE Standards

[SCTE 279] defines the operation of hardline amplifiers to be used 1.8 GHz broadband HFC networks. The use of the word Amplifier in this document refers to the amplifier described by [SCTE 279].

This document describes an information model for the [SCTE 279] amplifier. This document can be extended to other types of hardline amplifiers, for example, those used in 1.2 GHz broadband HFC networks.

6.2. Information models

Refer to [UML Guidelines] for information modeling concepts used throughout this specification.

6.2.1. Introduction

This specification uses the Unified Modeling Language (UML) to graphically represent the various elements composing the Information Model of an embedded amplifier module. The Information Model approach is based on object-oriented modeling 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 embedded amplifier.

The Information Models are divided into 3 sections:

1. Data type definitions including simple and complex data types.
2. Class Diagrams containing classes and their associations that define static information relevant to the feature. This also includes notifications related to the feature.
3. Component Diagrams defining operations on an interface that can be executed on the device (over the interface) for the feature.

Use Case diagrams illustrate the ways in which an external system can interact with the embedded amplifier. Sequence diagrams model the sequence of exchanged messages (e.g., SNMP, NETCONF, etc.) and actions for a specific operation and are often used to define specific scenarios for the defined Use Cases. The management information is represented in terms of classes or objects along with their attributes and the interactions between these encapsulated objects (or also referred to as entities in some representations). Class diagrams provide this conceptual, static model of the structure of the system. Class diagrams organize attributes into related groups or classes, model relationships between the classes including inheritance, and define the operations (e.g., read-only, read-write, CRUD) each class can execute.

Component diagrams model the components of a system and the interactions between the components. These define specific interfaces and the information that flows through the interface. Typically, one component provides the interface realization (the Server) while other component(s) utilize the interface (the Client(s)) to interact with the other component. Component diagrams effectively define interfaces and the operations which can be performed on the interface.

The collection of UML diagrams is referred to as the embedded amplifier information models. The managed objects, operations, and notifications are then represented in a protocol specific form referred to as a management data model. The management data models when using NETCONF are described using the YANG data modeling language [RFC 6020].

6.2.2. DataType Definitions

This section lists attribute type definitions used in the Amplifier Information Model. These attribute types are also described in [CM-OSSIv4.0]

Table 1 – UML Attribute Types

Attribute Type	Definition
AdminString	An octet string containing administrative information, preferably in human-readable form, limited to 255 bytes. The information is encoded as an octet string using the UTF-8 transformation format.
AdminStatusType	Reports the administrative status of an interface.
Boolean	Boolean has one of two possible values, usually denoted true and false.
Counter32	Counter32 specifies a value which represents a count. The range is 0 to 4,294,967,295. Counter32 does not have to start at zero. When a counter32 object reaches its maximum value of 4,294,967,295, the next value ***shall*** be zero.

DateTime	Used for values that contain both date and time parts and displays values in 'YYYY-MM-DD hh:mm:ss' format. The supported range is '1000-01-01 00:00:00' to '9999-12-31 23:59:59'.
Enum	A set of named values which are assigned identifiers that behave as constants in the language. The attribute can be assigned one of any of the enumerators as a value.
EnumBits	Allows selection of one or more of the enumerated attributes.
EventThrottleAdminStateType	This data type defines event throttling parameters.
EvPriorityType	This data type defines eight event priority levels. These are ordered from most critical (emergency) to least critical (debug).
EvReportingType	A bitmask that defines how an event is reported.
Float	A decimal64 floating point number as specified in [IEEE 754].
HexBinary	hex-encoded binary data and is a sequence of characters in which each binary octet is represented by two hexadecimal digits.
Host	Either a strongly-typed IP address or a FQDN. Use of this type avoids the weak validation inherent in the union-based inet:host type, as with this type an ip-address cannot be inappropriately validated as a domain-name accidentally. For a particular use of this data type, the eAMP *may* support only one of these choices: either an IP address or an FQDN.
Int	Whole numbers that range from -2,147,483,647 to 2,147,483,647. The number 2,147,483,648 is a reserved value and cannot be used.
IpAddress	an IP address (IPv4 or IPv6) and is either 4 bytes or 16 bytes.
IpAddressOriginType	Specifies how an IP address was assigned.
LatitudeType	A value lexically represented as digit degrees, minutes, seconds: ±DDMMSS.S
LongitudeType	A value lexically represented as digit degrees, minutes, seconds: ±DDDMMSS.S
MacAddress	An IEEE 802 media access control (MAC) address
MeasStatusType	This attribute reports the status of the PNM test instance.
OperStatusType	An Enum that defines the operational status of the entity.
String	An octet string containing administrative information, preferably in human-readable form. Lengths greater than 255 bytes are allowed, and the amplifier enforces the length limit. The information is encoded as an octet string using the UTF-8 transformation format.
TenthdB	power levels that are normally expressed in dB. Units are in tenths of a dB; for example, 5.1 dB will be represented as 51.
TenthdBmV	power levels that are normally expressed in dBmV. Units are in tenths of a dBmV; for example, 5.1 dBmV will be represented as 51.
TimeStamp	used for values that contain both date and time parts and has a range of '1970-01-01 00:00:01' UTC to '2038-01-19 03:14:07' UTC.
UnsignedByte	an integer value between 0 and 255, inclusive.
UnsignedInt	a 32-bit value that encodes a nonnegative integer in the range 0 to 4,294,967,295, inclusive.
UnsignedShort	a 16-bit value that encodes a nonnegative integer in the range of 0 and 65,535, inclusive.

6.3. Amplifier Reference Diagrams

The Figures in this section are examples and not intended to describe implementations. These examples can be used to show how the electronically configurable elements can be adjusted.

6.3.1. Signal Flow Diagram

The [SCTE 279] amplifier is frequency division duplex (FDD) where upstream and downstream signal flows use their own dedicated non-overlapping frequencies that are separated by diplex filters.

Figure 1 is a reference object diagram for an Amplifier with two ports.

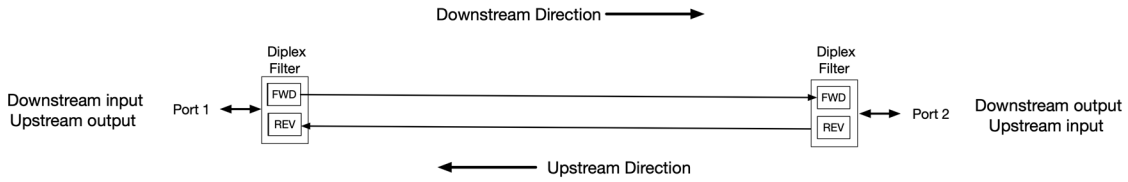

Figure 1 – Two-Port Amplifier Signal Flow Diagram

Port 1 is the downstream input and the upstream output. Likewise, port 2 is the downstream output and the upstream input.

Figure 2 is a reference object diagram for an Amplifier with 4 ports.

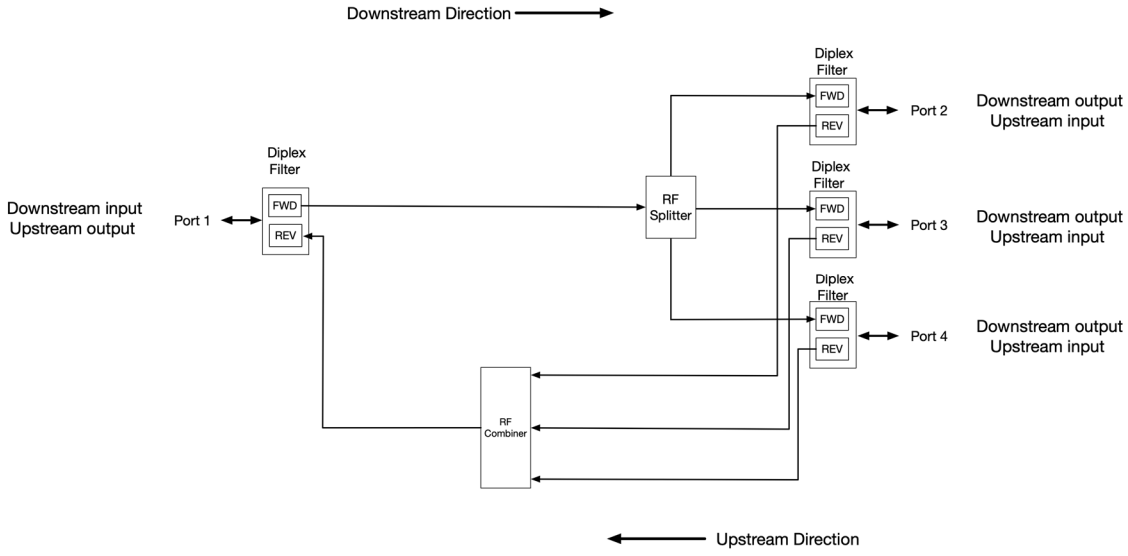

Figure 2 – Four-Port Amplifier Signal Flow Diagram

Port 1 is the downstream input and the upstream output. Port 2, 3 and 4 are downstream outputs and upstream inputs. It is possible for an amplifier to have more than 4 ports.

6.3.2. Configuration Diagrams

The figures in this section go along with the information in the RF information presented in Section 7.2, specifically capabilities and configuration.

Configuration in the downstream direction is referenced by the output port. That is, Port 1 will not have downstream capabilities or configuration.

Configuration in the upstream direction is referenced to the input port. That is, Port 1 will not have upstream capabilities or configuration.

Figure 3 is an example diagram of a two-port amplifier with multiple equalization (EQ), Attenuation (AT) stages. [SCTE 279] calls for the amplifier to have electronically controlled attenuation and equalization capability.

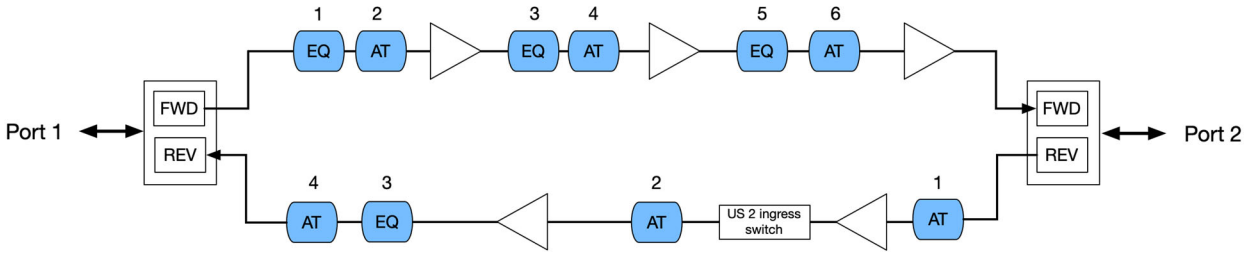

Figure 3 – Example Two-Port Amplifier Configuration Stages

The triangles represent gain stages however these are not electronically controlled with this information model.

In the downstream direction, with the input at Port 1 and the output at Port 2, in this example six numbered stages are shown that are electronically configurable: EQ, AT, EQ, AT, EQ and AT.

In the upstream direction, with the input at Port 2 and the output at Port 1, in this example there are four numbered stages that are electronically configurable: AT, AT, EQ, and AT.

The information model will show the capabilities of each stage as described in Section 7.2.1.6. Each stage is configurable as described in Section 7.2.3.8.

Also shown in Figure 3 is the upstream ingress switch which is described in [SCTE 279]. These capabilities are described in this document in Section 7.2.1.4 and the configuration is described in Section 7.2.3.5.

Figure 4 shows just the downstream direction and includes the capabilities objects for each electronically configurable stage. Configuration of these stages would have to use values within the available range of either attenuation or equalization.

For example, stage 1 is equalization. The location is specific to the product, and this stage is advertised as input. Note that stages 3 and 4 are advertised as interstage and stages 5 and 6 are advertised as output. Since stage 1 is equalization, the values for minAttenuation, maxAttenuation, and AttenuationStepSize are not used and can be set to zero. The values of minEqualization, maxEqualization and

EqualizationStepSize describe how the value of equalization can be configured. In this example, minEqualization = -25.0 dB, maxEqualization = 25.0 dB, and the EqualizationStepSize is 0.5 dB. When configured, the value of equalization for this stage can be anywhere between -25.0 dB and 25.0 dB in steps of 0.5 dB.

Similarly, the capabilities for stages 2 through 6 are also shown in Figure 4.

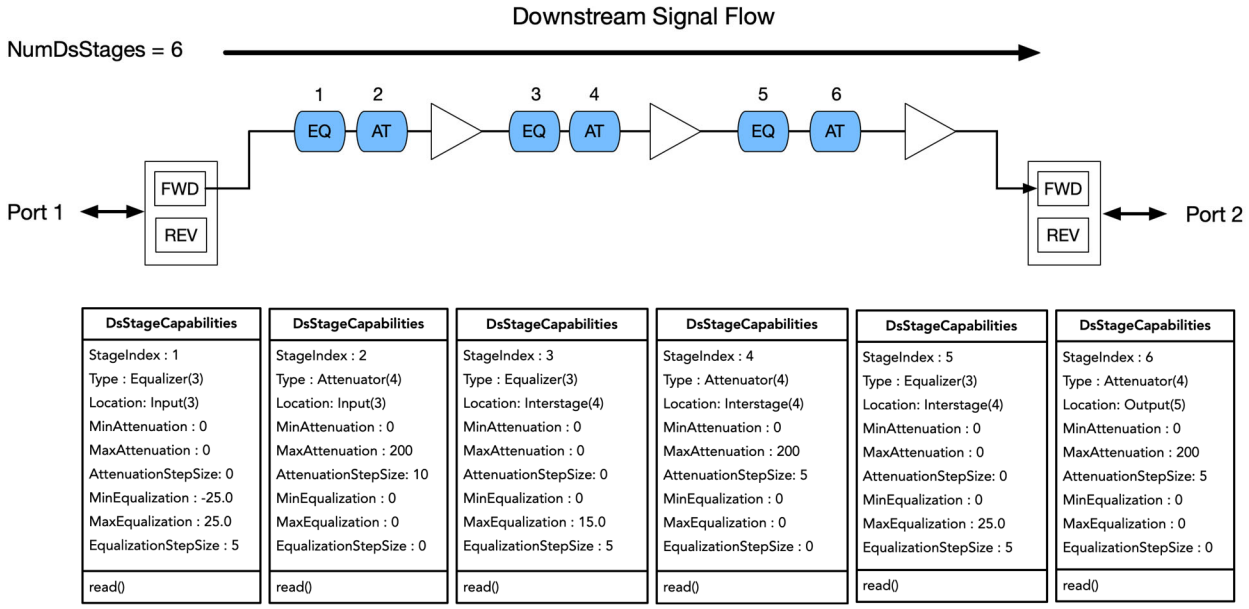

Figure 4 – Example Two-Port Amplifier Downstream Stages

Figure 5 shows just the upstream direction and includes the capabilities objects for each electronically configurable stage. Configuration of these stages would use values within the available range of either attenuation or equalization for the specific stage.

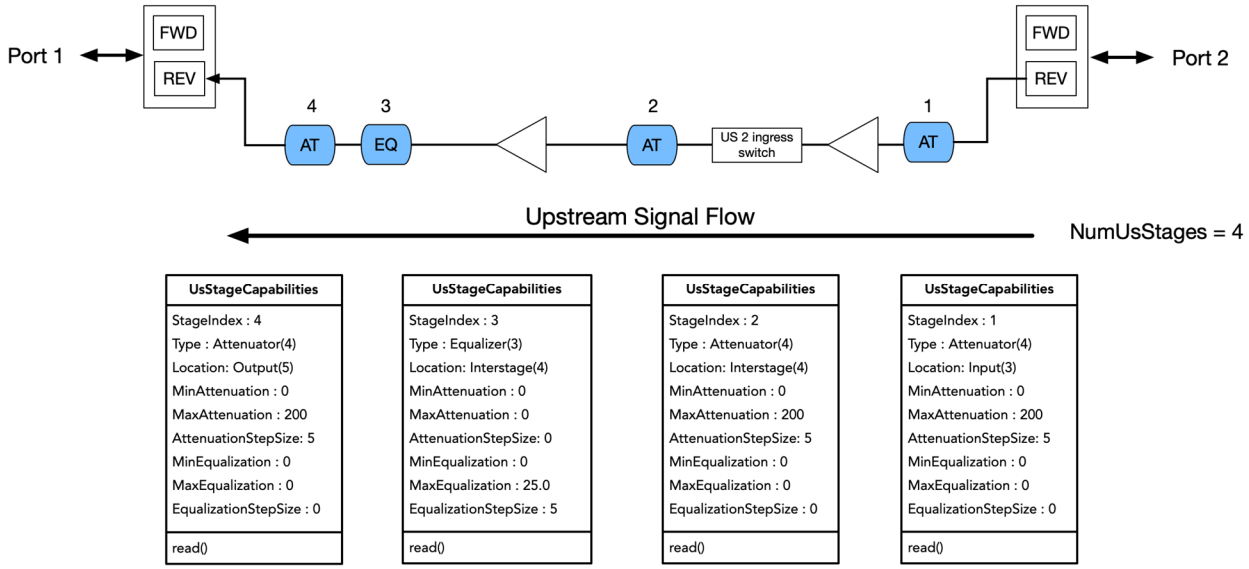

Figure 5 – Example Two-Port Amplifier Upstream Stages

Figure 6 shows example electronically configurable stages for a four-port amplifier, that has 3 output ports and is sometimes referred to as a balanced triple amplifier. This example will illustrate why knowledge of the internals of the amplifier is needed to support operations.

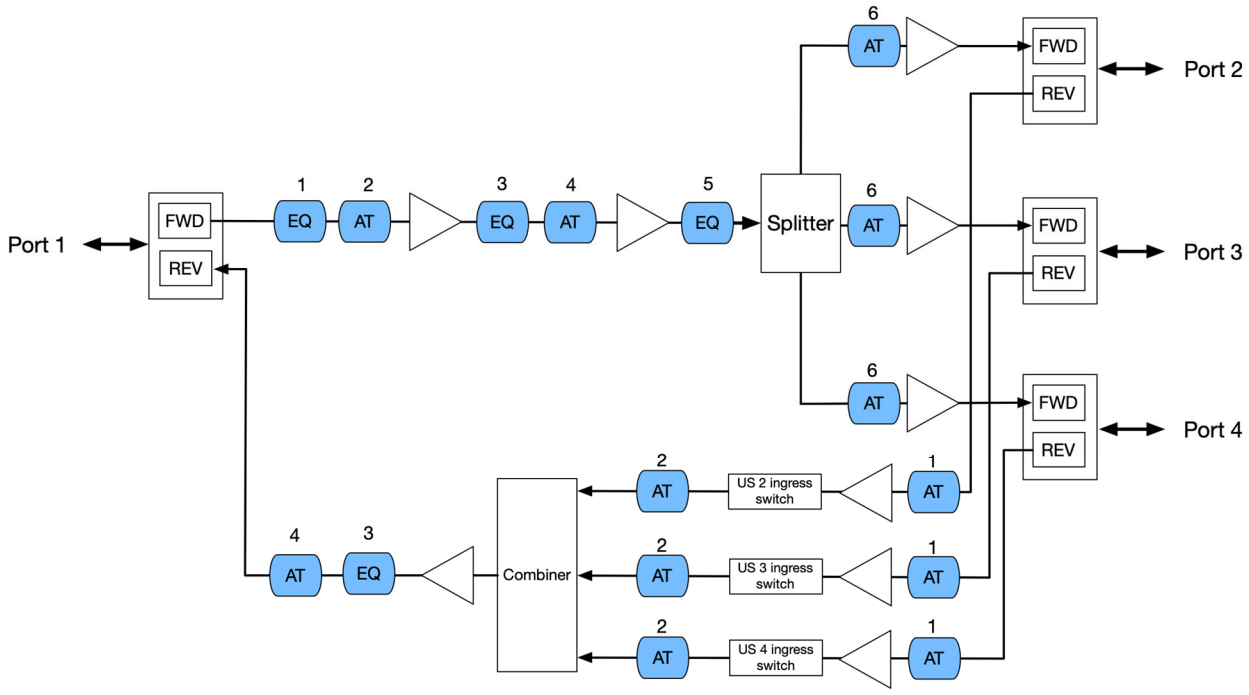

Figure 6 – Example Balanced Triple Amplifier

Note that Figure 6 has three downstream output ports (Port 2, Port 3, and Port 4) that share the first five electronically configurable stages. For the output ports, only stage six is unique per output port. This means that adjusting any of the first five stages will affect all output ports.

Similarly in the upstream direction, there are three upstream input ports (Port 2, Port 3, and Port 4) and each has four electronically configurable stages. For the input ports, stages one (AT) and two (EQ) are unique per port and stages three (EQ) and four (AT) are shared by all upstream ports (after the combiner) and adjustments to stages three and four will affect the upstream output at Port 1.

The internal configurations of an amplifier are unique, and knowledge of the available electronically configurable stages and which outputs are affected are important.

6.3.3. Object Diagram

Figure 7 is a reference object diagram for the Amplifier.

Physical port 1 is both the downstream input to the amplifier and the upstream output of the amplifier.

Physical port 2 (and up) is both a downstream output and an upstream input to the amplifier.

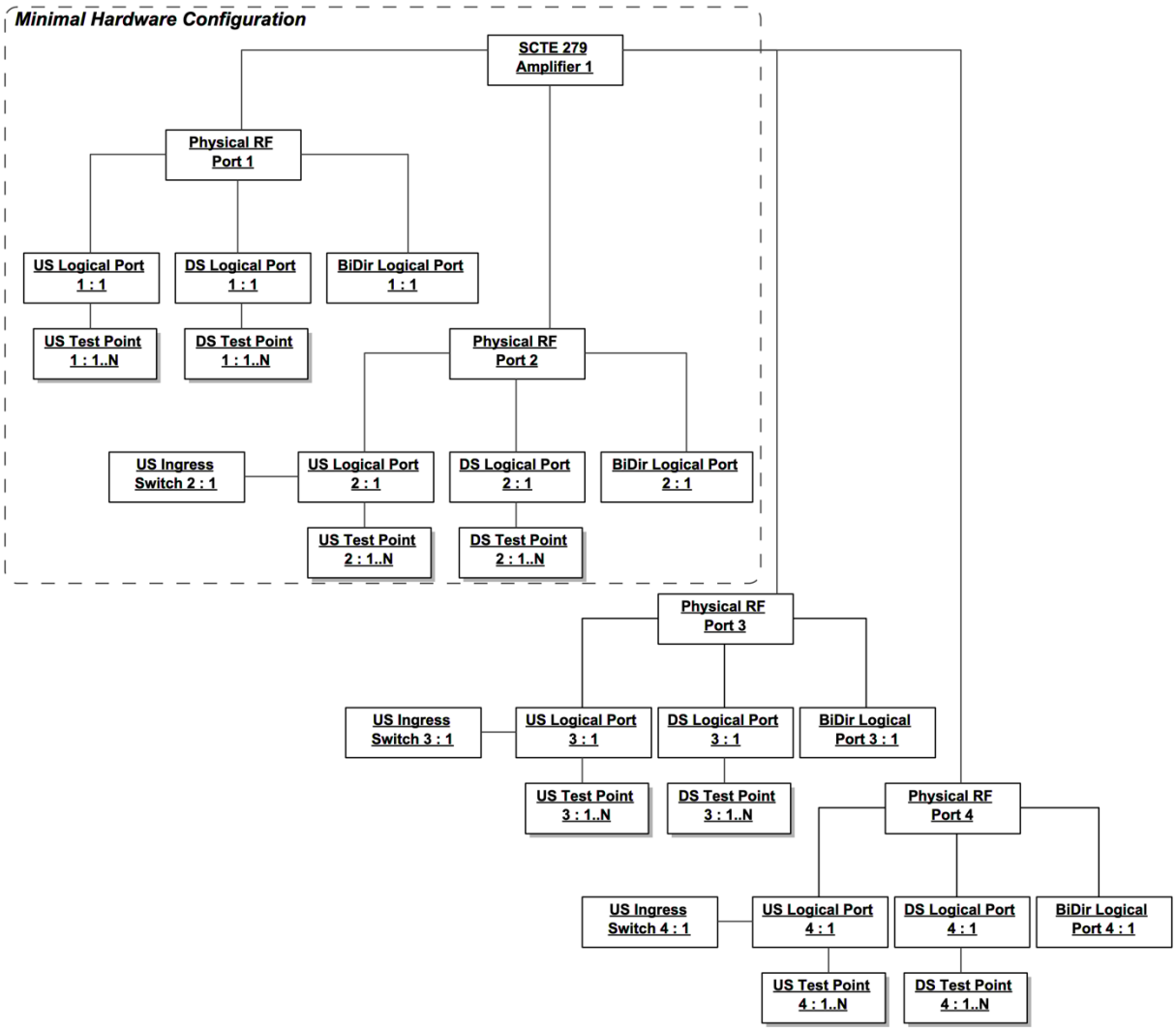

Figure 7 – Amplifier Object Diagram

The Amplifier has both physical and logical ports. A physical port is composed of:

An upstream logical port for management of upstream objects, for example, the upstream ingress switch.

A downstream logical port for management of downstream objects, for example, automatic gain control (AGC).

A bi-directional logical port for management of bi-directional objects, for example, the diplex filter associated with that physical port.

6.4. Transponder

This standard does not place requirements on the transponder.
Figure 8 shows a logical model of the Amplifier with both a transponder and embedded Amplifier (eAMP) logical entities. Both the local management interface described in [SCTE 279] and the information model in this document are associated with the eAMP.

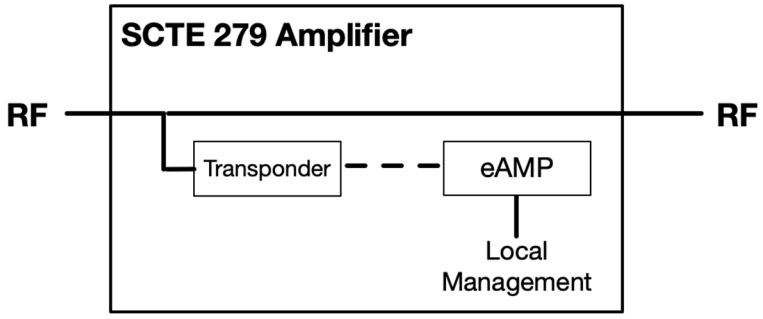

Figure 8 – Amplifier Logical Model

The specification neither describes the operation of the transponder nor places requirements on the transponder; however, is necessary for two-way communications with the Amplifier.

The eAMP logical entity contains the embedded Amplifier functions described by the information model in this document.

While there is no requirement the transponder be a DOCSIS cable modem, as an example, Figure 9 shows the Amplifier using an embedded DOCSIS cable modem (eCM) as a transponder. The local management interface described in [SCTE 279] is associated with the eAMP. In the example the transponder is a DOCSIS cable modem, there are considerations of the DOCSIS version, the associated network management tools of that modem and how that modem is internally connected to the amplifier.

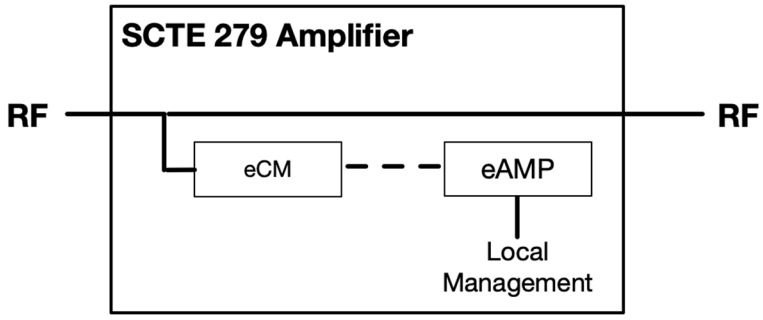

Figure 9 – Amplifier Logical Model with Embedded DOCSIS Cable Modem

Using a DOCSIS cable modem as the transponder is described by the embedded DOCSIS specification described in [eDOCSIS].

7. Amplifier Information Model

Figure 10 defines the top-level Amplifier related classes and signals (notifications).

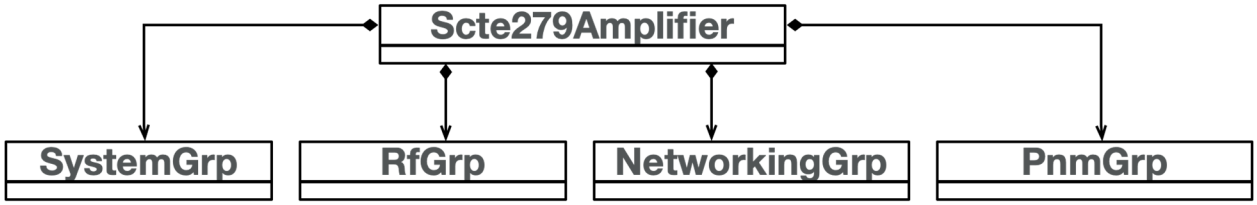

Figure 10 – Amplifier Top Level Class Diagram

The System Group is discussed in Section 7.1. This information is used to describe the amplifier including items like supplier, model, location and hardware and software versions. As includes is information on sensors associated with the amplifier and the power supply.

The RF Group is discussed in Section 7.2. This information is used to describe the RF capabilities of the amplifier, and the RF configuration of the amplifier.

The Networking Group is discussed in Section 7.3. These are read-only objects that describe network servers the amplifier is aware of.

The PNM Group is discussed in Section 7.4. This information can be used to gather diagnostic information on the underlying coaxial cable network. The PNM Group aligns with data collection already described in DOCSIS specifications and introduces the spectrum capture measurement to the Amplifier.

7.1. System Group (SystemGrp)

Figure 11 defines the system Group related classes and signals. These classes have read-only attributes.

The Amplifier ***shall*** implement all classes as described in the SystemGrp information model.

7.1.1. System Status Group

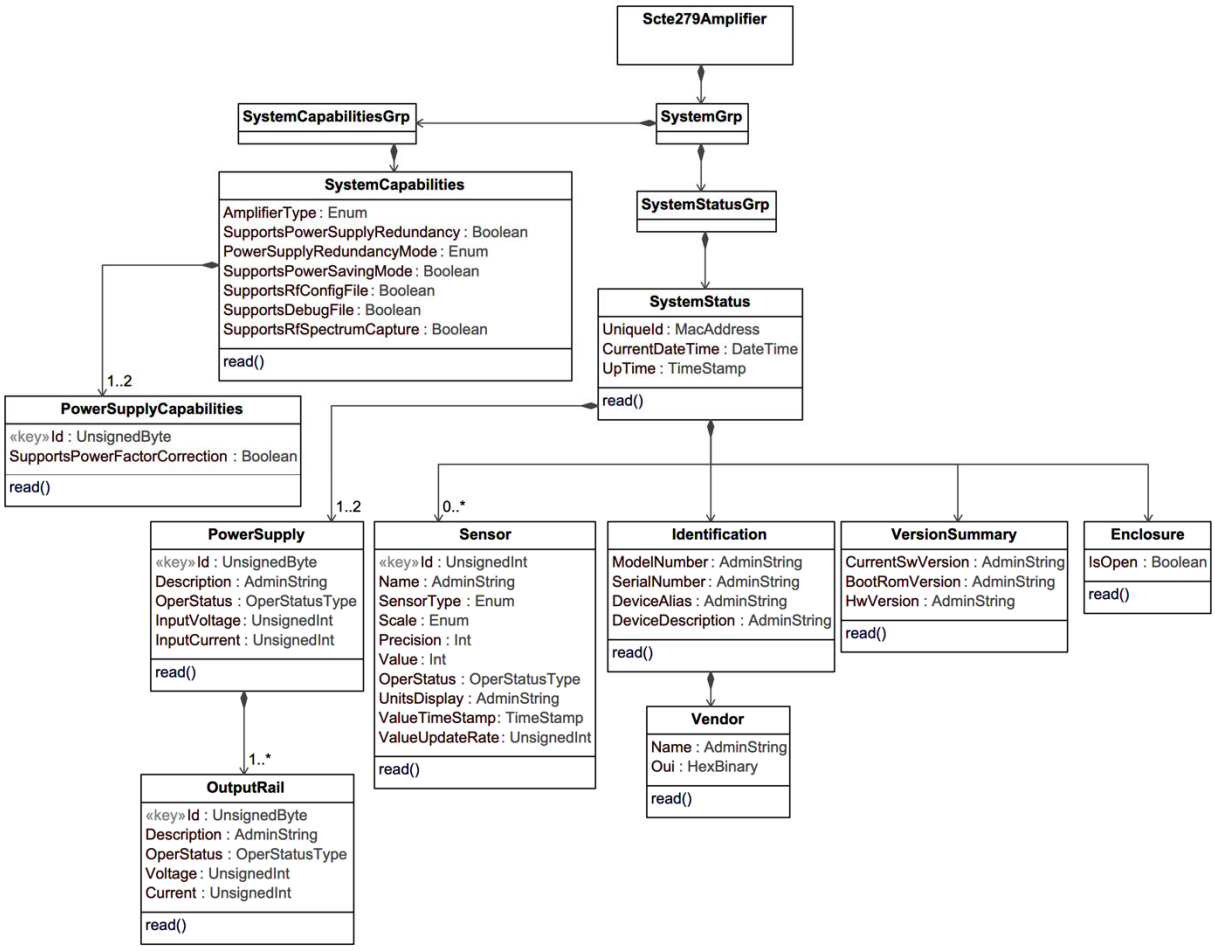

Figure 11 – System Status Class Diagram

7.1.1.1. SystemCapabilities

This Class lists system capabilities of the Amplifier.

Table 2 – SystemCapabilities Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
AmplifierType	Enum	ro	other(1), unknown(2), multiport(3), lineExtender(4), booster(5), compact(6)		
SupportsPowerSupplyRedundancy	Boolean	ro			
PowerSupplyRedundancyMode	Enum	ro	other(1), unknown(2), loadSharing(3), switchedRedundant(4)		

Attribute Name	Type	Access	Type Constraints	Units	Default Value
SupportsPowerSavingMode	Boolean	ro			
SupportsRfConfigFile	Boolean	ro			
SupportsDebugFile	Boolean	ro			
SupportsRfSpectrumCapture	Boolean	ro			

7.1.1.1.1. AmplifierType

SP920 defines four types of amplifiers including Multiport, Line Extender, Booster and Compact amplifiers.

7.1.1.1.2. SupportsPowerSupplyRedundancy

Identifies if the Amplifier supports power supply redundancy.

7.1.1.1.3. PowerSupplyRedundancyMode

If Power Supply redundancy is supported, this item identifies the type of power supply redundancy.

7.1.1.1.4. SupportsPowerSavingMode

Indicates if the Amplifier supports features to reduce power consumption

7.1.1.1.5. SupportsRfConfigFile

Identifies if the Amplifier supports an RF configuration file that the operator can download. See Section 7.1.9 for additional details.

The Amplifier *may* implement the RF configuration file.

7.1.1.1.6. SupportsDebugFile

Identifies if the Amplifier supports a debug file that the operator can download. See Section 7.1.9 for additional details.

The Amplifier *may* implement the debug file.

7.1.1.1.7. SupportsRfSpectrumCapture

Identifies if the Amplifier supports a RF Spectrum Capture to a file that the operator can download. See Section 7.1.9 for additional details.

The Amplifier *may* implement RF spectrum capture.

7.1.1.2. PowerSupplyCapabilities

This Class lists power supply capabilities.

Table 3 – PowerSupply Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Id	UnsignedByte	ro	1, 2		
SupportsPowerFactorCorrection	Boolean	ro			

7.1.1.2.1. Id

The Id is the unique identifier of the power supply on the Amplifier.

7.1.1.2.2. SupportsPowerFactorCorrection

Indicates if the power supply supports power factor correction. True indicates the power supply supports power factor correction. False indicates the power supply does not support power factor correction.

7.1.2. System Status Group (SystemStatusGrp)

7.1.2.1. SystemStatus

The SystemStatus object lists key status items.

Table 4 – SystemStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
UniqueId	MacAddress	ro		N/A	
CurrentDateTime	DateTime	ro			
UpTime	Timestamp	ro		Hundredths of a second	

7.1.2.1.1. UniqueID

This attribute specifies a globally unique 6-byte MAC Address identifier for the Amplifier.

7.1.2.1.2. CurrentDateTime

The current system date and time.

7.1.2.1.3. UpTime

The time in hundredths of a second since the network management portion of the Amplifier was last re-initialized.

7.1.2.2. Identification

The Identification object identifies the Amplifier.

Table 5 – Identification Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
ModelNumber	AdminString	ro			
SerialNumber	AdminString	ro			
DeviceAlias	AdminString	ro			
DeviceDescription	AdminString	ro			

7.1.2.2.1. ModelNumber

This attribute reports the model's name and number used by the vendor to identify the Amplifier. The format is vendor specific.

7.1.2.2.2. SerialNumber

This attribute reports the serial number of the Amplifier. The format is vendor specific.

7.1.2.2.3. DeviceAlias

This attribute reports the device name assigned by the operator and represents a "handle" for the Amplifier.

7.1.2.2.4. DeviceDescription

This attribute reports a short text description of the Amplifier provided by the manufacturer.

7.1.2.3. Vendor

The Vendor object identifies the supplier of the Amplifier.

Table 6 – Vendor Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Name	AdminString	ro			
Oui	HexBinary	ro	SIZE(3)		

7.1.2.3.1. Name

This attribute reports the vendor's name of the Amplifier. The format is vendor specific.

7.1.2.3.2. Oui

The vendor organizationally unique identifier (vendor OUI) is a 3-byte hexbinary string that contains the IEEE Company Identifier for the vendor. A value of all zero in the 3-byte string indicates that the vendor OUI is unspecified.

7.1.2.4. VersionSummary

The VersionSummary object identifies software and hardware versions of the Amplifier.

Table 7 – VersionSummary Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
CurrentSwVersion	AdminString	ro			
BootRomVersion	AdminString	ro			
HwVersion	AdminString	ro			

7.1.2.4.1. CurrentSwVersion

This attribute reports the version number of the software currently running on the Amplifier. The format is vendor specific.

7.1.2.4.2. BootRomVersion

This attribute reports the version number of the boot ROM currently installed on the Amplifier. The format is vendor specific.

7.1.2.4.3. HwVersion

This attribute reports the revision number of the Amplifier hardware. The format is vendor specific.

7.1.2.4.4. Enclosure

Identifies if the enclosure for the Amplifier is open or closed.

Table 8 – Enclosure Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
IsOpen	Boolean	ro			

7.1.2.4.5. IsOpen

If a sensor indicates the lid is open, this object is True. If a sensor indicates the lid is closed, this object is False.

The Amplifier ***shall*** implement a sensor to indicate if the lid is open or closed.

7.1.2.5. Sensor

This module defines sensor status objects, a set of generic fields for transport of Amplifier generalized metric information such as temperatures, light, and other types of sensors.

Table 9 – Sensor Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Id	UnsignedInt	ro			
Name	AdminString	ro			
SensorType	Enum	ro	other(1), unknown(2), voltsAC(3), voltsDC(4), amperes(5), watts(6), hertz(7), celsius(8), percentRH(9), rpm(10), cmm(11), truthvalue(12)		
Scale	Enum	ro	yocto(1), zepto(2), atto(3), femto(4), pico(5), nano(6),		

Attribute Name	Type	Access	Type Constraints	Units	Default Value
			micro(7), milli(8), units(9), kilo(10), mega(11), giga(12), tera(13), exa(14), peta(15), zetta(16), yotta(17)		
Precision	Int	ro	-8..9		
Value	Int	ro	-1000000000.. 1000000000		
OperStatus	OperStatusType	ro	other(0), unknown(1), up(2), down(3)		
UnitsDisplay	AdminString	ro			
ValueTimeStamp	TimeStamp	ro			
ValueUpdateRate	UnsignedInt	ro		milliseconds	

7.1.2.5.1. Id

The sensor id is the unique identifier of the sensor on the Amplifier.

7.1.2.5.2. Name

The sensor name is the human readable name for the sensor.

7.1.2.5.3. SensorType

The type of data returned by the associated entPhySensorValue object. The SensorType object *should* be set by the agent during entry creation. The value of the SensorType object *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).

7.1.2.5.4. Scale

This attribute reports a data scaling factor, represented with an International System of Units (SI) prefix. The actual data units are determined by examining this attribute together with the associated SensorType attribute. This attribute is defined together with SensorType and Precision.

Together, the three attributes (Scale, SensorType and Precision) are used to identify the semantics of the Value attribute.

The Scale object *should* be set by the agent during sensor entry creation.
The value of the Scale object *should not* change during senor operation.

YOCTO The International System of Units for 10e-24.
ZEPTO The International System of Units for 10e-21.
ATTO The International System of Units for 10e-18.
FEMTO The International System of Units for 10e-15.
PICO The International System of Units for 10e-12.
NANO The International System of Units for 10e-9.
MICRO The International System of Units for 10e-6.
MILLI The International System of Units for 10e-3.
UNITS Units for 10e0.
KILO The International System of Units for 10e3.
MEGA The International System of Units for 10e6.
GIGA The International System of Units for 10e9.
TERA The International System of Units for 10e12.
EXA The International System of Units for 10e18.
PETA The International System of Units for 10e15.
ZETTA The International System of Units for 10e21.
YOTTA The International System of Units for 10e24.

7.1.2.5.5. Precision

If an object of this type contains a value in the range 1 to 9, it represents the number of decimal places in the fractional part of an associated EntitySensorValue fixed-point number.

If an object of this type contains a value in the range -8 to -1, it represents the number of accurate digits in the associated EntitySensorValue fixed-point number.

The value zero indicates the associated EntitySensorValue object is not a fixed-point number.

Agent implementors choose a value for the associated EntitySensorPrecision object so that the precision and accuracy of the associated EntitySensorValue object is correctly indicated. For example, a physical entity representing a temperature sensor that can measure 0 degrees to 100 degrees C in 0.1 degree increments, +/- 0.05 degrees, would have an EntitySensorPrecision value of ’1’, an EntitySensorDataScale value of ’units(9)’, and an EntitySensorValue ranging from ’0’ to ’1000’. The EntitySensorValue would be interpreted as ’degrees C * 10’.
The Precision object *should* be set by the agent during entry creation. The Precision object *should not* change during operation.

7.1.2.5.6. Value

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

To correctly interpret the value of this object, the associated entPhySensorType, entPhySensorScale, and entPhySensorPrecision objects ***shall*** also be examined.

An object using this data type represents a Sensor value. An object of this type is defined together with objects of type SensorType, SensorScale and SensorPrecision. Together, associated objects of those three

types are used to identify the semantics of an object of this data type. The semantics of an object using this data type are determined by the value of the associated SensorType object. If the associated SensorType object is equal to ‘voltsAC(3)’, ‘voltsDC(4)’, ‘amperes(5)’, ‘watts(6)’, ‘hertz(7)’, ‘celsius(8)’, or ‘cmm(11)’, then an object of this type contains a fixed point number ranging from -999,999,999 to +999,999,999.

The value -1000000000 indicates an underflow error. The value +1000000000 indicates an overflow error.

The SensorPrecision indicates how many fractional digits are represented in the associated SensorValue object.

If the associated SensorType object is equal to ‘percentRH(9)’, then an object of this type contains a number ranging from 0 to 100.

If the associated SensorType object is equal to ‘rpm(10)’, then an object of this type contains a number ranging from -999,999,999 to +999,999,999.

If the associated SensorType object is equal to ’truthvalue(12)’, then an object of this type contains either the value ‘true(1)’ or the value ‘false(2)’.

If the associated SensorType object is equal to ‘other(1)’ or ‘unknown(2)’, then an object of this type contains a number ranging from -1000000000 to 1000000000.

7.1.2.5.7. OperStatus

This attribute reports the current operational status of the sensor.

other(0) the status is a supplier-specific state.
unknown(1) the status is unknown.
up(2) the status is operational.
down(3) the status is not operational with no further information.

7.1.2.5.8. UnitsDisplay

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

7.1.2.5.9. ValueTimeStamp

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

7.1.2.5.10. ValueUpdateRate

An indication of the frequency that the agent updates the associated physical-sensor-value object, representing in milliseconds. The value zero indicates any of

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.

7.1.2.6. PowerSupply

This module identifies the power supply in the Amplifier, and the input voltage and current.

Table 10 – PowerSupply Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Id	UnsignedByte	ro			
Description	AdminString	ro			
OperStatus	OperStatusType	ro	other(0), unknown(1), up(2), down(3)		
InputVoltage	UnsignedInt	ro		0.1 volt RMS	
InputCurrent	UnsignedInt	ro		0.1 amp RMS	

7.1.2.6.1. Id

The id is the unique identifier of the power supply on the Amplifier.

7.1.2.6.2. Description

Contains the description of the power supply.

7.1.2.6.3. OperStatus

This attribute reports the current operational status of the power supply.

other(0) - the status is a supplier-specific state.
unknown(1) - the status is unknown.
up(2) - the status is operational.
down(3) - the status is not operational with no further information.

7.1.2.6.4. InputVoltage

Describes the input voltage at the power supply. This value is reported in tenth of Volts RMS, for example, a value of 1200 means 120.0 Volts RMS.

A value of zero indicates the InputVoltage is not known.
If InputVoltage is not supported by the power supply, it will return a value of zero.

7.1.2.6.5. InputCurrent

Describes the input current at the power supply. This value is reported in tenth of Amps RMS, for example, a value of 100 means 10.0 Amps RMS.

A value of 0 indicates the InputCurrent is not known.
If InputCurrent is not supported by the power supply, it will return a value of zero.

7.1.2.7. Output Rail

This module identifies the output power rails. If the power rail provides direct current the units are volts and amps. If the power rail provides alternating current, the units ae in volts RMS and amps RMS.

Table 11 – OutputRail Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Id	UnsignedByte	ro			
Description	AdminString	ro			
OperStatus	OperStatusType		other(0), unknown(1), up(2), down(3)		
Voltage	UnsignedInt	ro		0.1 volt	
Current	UnsignedInt	ro		0.1 amp	

7.1.2.7.1. Id

The id is the unique identifier of the output rail on the Amplifier.

7.1.2.7.2. Description

Contains the description of the output rail.

7.1.2.7.3. OperStatus

This attribute reports the current operational status of the power rail.

other(0) - the status is a supplier-specific state.
unknown(1) - the status is unknown.
up(2) - the status is operational.
down(3) - the status is not operational with no further information.

7.1.2.7.4. Voltage

Describes the output of the power rail. This value is reported in tenth of volts, for example, a value of 242 means 24.2 volts.

A value of zero indicates the Voltage is not known.
If Voltage is not supported by the power supply, it will return a value of zero.

7.1.2.7.5. Current

Describes the output of the power rail. This value is reported in tenth of amperes, for example, a value of 22 means 2.2 amperes.

A value of 0 indicates the Current is not known.
If Current is not supported by the power supply, it will return a value of zero.

7.1.3. System Configuration Group (SystemCfgGrp)

7.1.3.1. System Configuration (SystemCfg)

Figure 12 defines the system configuration related classes and signals. These are classes have read-write attributes.

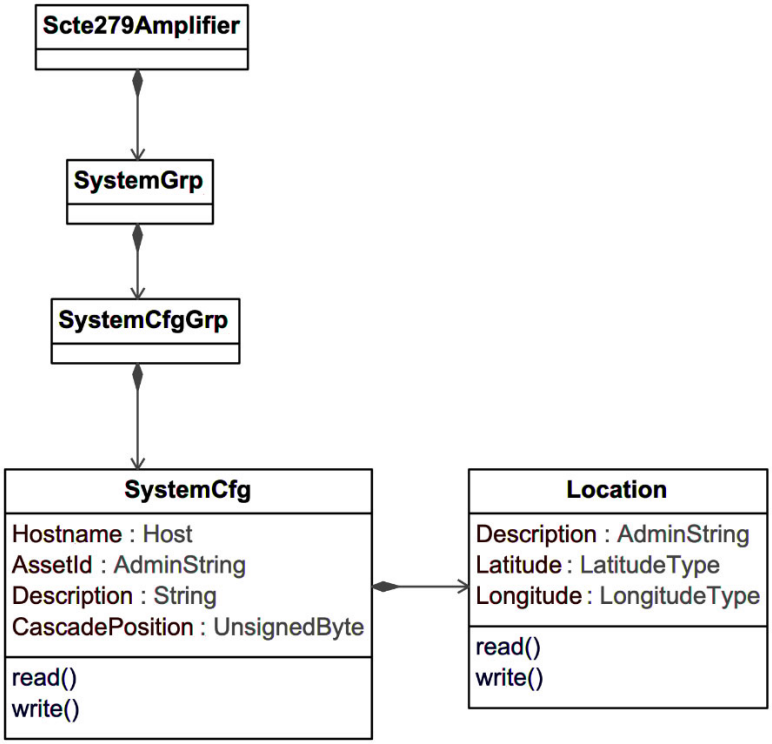

Figure 12 – System Configuration Class Diagram

7.1.3.2. SystemCfg

The SystemCfg element is the root of the configuration and state object tree. It represents a single Amplifier.

Table 12 – SystemCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Hostname	Host	rw			
AssetId	AdminString	rw			
Description	String	rw			
CascadePosition	UnsignedByte	rw			

7.1.3.2.1. Hostname

This attribute specifies the IP Host for the eAMP.

7.1.3.2.2. AssetId

A short string used to identify this asset. The value and meaning of the AssetId is left to the operator.

The value of AssetId ***shall*** be stored in non-volatile memory and persist across reboots.

7.1.3.2.3. Description

This attribute specifies a human-readable description for the Amplifier. The Value and meaning of the description are left to the operator.

The value of Description ***shall*** be stored in non-volatile memory and persist across reboots.

7.1.3.2.4. CascadePosition

This attribute specifies the location of the amplifier in the cascade. For example if the amplifier is the first after the fiber node, then the value is 1.

The value of CascadePosition ***shall*** be stored in non-volatile memory and persist across reboots.

7.1.3.3. Location

The Location object identifies the location of the Amplifier.

Table 13 – Location Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Description	AdminString	rw			
Latitude	LatitudeType	rw	SIZE(9)		
Longitude	LongitudeType	rw	SIZE(10)		

7.1.3.3.1. Description

This attribute reports the location of the Amplifier.

7.1.3.3.2. Latitude

This attribute allows the latitude portion of the Amplifier's geographic location to be written to the Amplifier. It consists of a 9-byte string formatted as specified in [ISO 6709]. The Amplifier uses '6-digit notation' in the format: deg, min, sec, ±DDMMSS.S (for example: -750015.1).

There is no requirement the Amplifier implement a global positioning system (GPS) receiver. Latitude could be entered either locally by a technician during initial setup or remotely using network management.

A value of '+000000.0' indicates that the latitude is not set.

The value of Latitude ***shall*** be stored in non-volatile memory and persist across reboots.

7.1.3.3.3. Longitude

This attribute allows the longitude portion of the Amplifier's geographic location to be written to the Amplifier. It consists of a 10-byte long string formatted as specified in [ISO 6709]. The Amplifier uses '7-digit notation' in the format: deg, min, sec, ±DDDMMSS.S (for example: -0100015.1).

There is no requirement the Amplifier implement a global positioning system (GPS) receiver. Longitude could be entered either locally by a technician during initial setup or remotely using network management.

A value of +0000000.0 indicates that the longitude is not set.

The value of Longitude ***shall*** be stored in non-volatile memory and persist across reboots.

7.1.4. Reset Status Group (ResetStatusGrp)

Figure 13 defines the reset status related classes and signals. These are classes have read-only attributes.

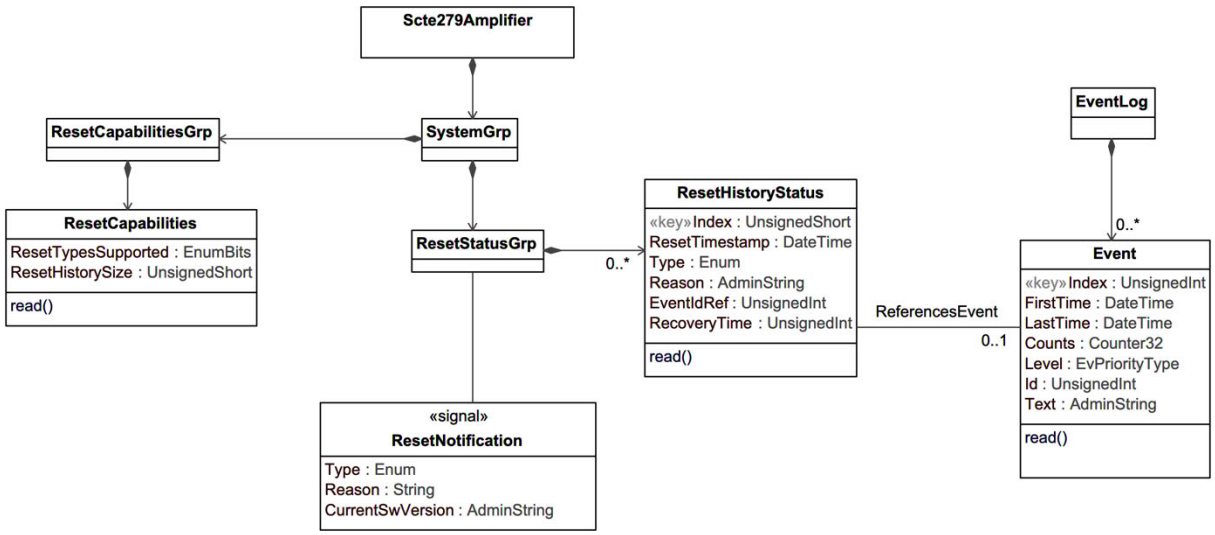

Figure 13 – Reset Status Class Diagram

7.1.4.1. ResetCapabilities

ResetCapabilities object reports the capabilities to support various types of resets.

Table 14 – ResetCapabilities Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
ResetTypesSupported	EnumBits	ro	softReset(1), hardReset(2), nvReset(3), factoryReset(4)		
ResetHistorySize	UnsignedShort	ro			

7.1.4.1.1. ResetTypesSupported

This defines the possible reset types. Reset types are described in Section 7.1.5.2.

softReset(1): The Amplifier performed a soft reset.

hardReset(2): The Amplifier performed a power-on reset or equivalent reset.

nvReset(3): The Amplifier cleared most non-volatile configuration and performed a hard reset.

factoryReset(4): The Amplifier set all settings back to original factory settings and performed a hard reset.

7.1.4.1.2. ResetHistorySize

This attribute provides the maximum size of the ResetHistory object.

The Amplifier ***shall*** support a minimum of 5 entries in the ResetHistoryStatus list.

7.1.4.2. ResetHistoryStatus

ResetHistoryStatus object provides a record of the occurrences of a reset of an Amplifier. The Amplifier can reset for many reasons, including power failure, hardware, or software failure, or reset command from an operations system. This object provides details of the conditions that caused the Amplifier to reset to assist with understanding why a reset occurred. This object also provides details on how much time it took for the Amplifier to boot and to become operational.

Once the Amplifier is operational, the Amplifier ***shall*** include the most recent reset instance in the reset occurrences.

The Amplifier ***shall*** store a record of each reset occurrence in non-volatile storage when the Amplifier resets/reboots for any reason.

Table 15 – ResetHistoryStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Index	UnsignedShort	key			
ResetTimestamp	DateTime	ro			
Type	Enum	ro	softReset(1), hardReset(2), nvReset(3), factoryReset(4)		
Reason	AdminString	ro			
EventIdRef	UnsignedInt	ro			
RecoveryTime	UnsignedInt	ro			

7.1.4.2.1. Index

This attribute is a key and provides an index into the table. Indexes are consecutively ordered from 0..N-1 where N is the maximum number of reset history events supported by the Amplifier as reported by the ResetHistorySize attribute of the ResetCapabilities object. This attribute corresponds to the order in which reset events occurred. The most recent reset event index has a value of zero.

7.1.4.2.2. ResetTimestamp

This attribute provides the time at which the reset occurred.

If the Amplifier does not know what time the Amplifier reset occurred, the Amplifier *may* report a value of midnight, Jan 1, 1970 UTC in the ResetTimestamp attribute.

7.1.4.2.3. Type

This attribute identifies the type of reset that the Amplifier is reporting. The possible values are described in Section 7.1.5.2.

7.1.4.2.4. Reason

This attribute provides a vendor specific string with a detailed explanation of why the Amplifier was reset. Vendors are expected to add additional details, including diagnostic information, whether this was a normal hard reset vs. a vendor specific type of hard reset, etc.

7.1.4.2.5. EventIdRef

If the reset was related to an event that was captured in a standardized Amplifier event, this attribute provides the event ID of the corresponding event. Events are described in Section 7.1.6.

If the Amplifier does not generate a standard event corresponding to the reset occurrence, this attribute will have a value of 0.

7.1.4.2.6. RecoveryTime

This attribute provides the number of seconds that have elapsed between the time the reset event started and the completion of the local Amplifier initialization stage. The calculation of the number of seconds in this period is vendor specific.

The Amplifier might not always be able to record the timestamp of the most recent reset event. In addition, the Amplifier might not be able to determine the time of the completion of the local initialization stage. For example, after a reset event, the Amplifier could reset again before being able to obtain the current time.

7.1.4.3. ResetNotification

ResetNotification object provides a notification of a reset of an Amplifier.

Table 16 – ResetNotification Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Type	Enum		softReset(1), hardReset(2), nvReset(3), factoryReset(4)		
Reason	AdminString				
CurrentSwVersion	AdminString				

7.1.4.3.1. Type

This attribute identifies the type of reset that the Amplifier is recording.

7.1.4.3.2. Reason

This attribute provides a vendor specific string with a detailed explanation of why the Amplifier was reset. Vendors are expected to add additional details, including diagnostic information, whether this was a normal hard reset or a vendor specific type of hard reset, etc.

7.1.4.3.3. CurrentSwVersion

This attribute identifies the current software version when the Amplifier was reset, and is the same value stored under CurrentSwVersion in the System Status Group.

7.1.5. Reset Control Component Diagram

Figure 14 shows the ResetControl component diagram and illustrates the Amplifier and Network Management application components that provide reset control operations on the Amplifier.

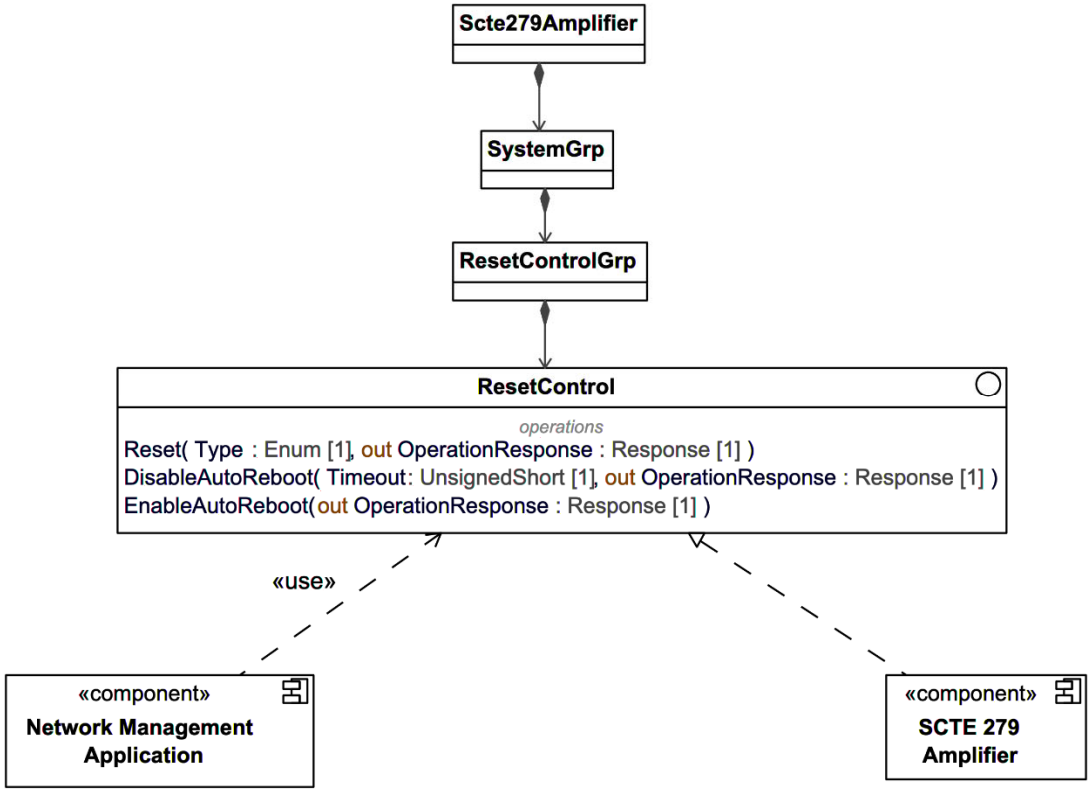

Figure 14 – Reset Management Component Diagram

7.1.5.1. ResetControl

The ResetControl interface manages operations associated with the reset of the Amplifier. Operations include:

- Reset
- DisableAutoReboot
- EnableAutoReboot

The Amplifier might reboot automatically if it is not able to successfully complete the initialization. The DisableAutoReboot operation disables automatic reboot of the Amplifier for a specified period to allow remote connection that is uninterrupted by an automatic reboot. If an Amplifier is unable to initialize and is stuck in a cycle of automatic reboots, this operation allows the automatic reboot to be disabled so that the debugging process is not interrupted by automatic reboot. DisableAutoReboot has a timeout parameter so that an Amplifier is not accidentally kept from rebooting in the future.

The Amplifier ***shall*** implement the ResetControl operations.

Table 17 – ResetControl Operations

Operation Name	Type	Access	Type Constraints	Units	Default Value
Reset	Enum	rw	softReset(1), hardReset(2), nvReset(3), factoryReset(4)		
DisableAutoReboot	UnsignedShort	rw	1..360	Seconds	360
EnableAutoReboot		rw			

7.1.5.2. Reset Operation

This operation allows a reset to be commanded of the Amplifier. Reading this operation always returns a value of softReset.

Reset represents an asynchronous operation that takes as an input parameter the reset type. As a successful result of the operation, the Amplifier is reset. There are no output parameters to this operation.

Table 18 – Reset Operation Parameters

Parameter Name	Type	Type Constraints	Direction	Multiplicity	Units
Type	DateTime		In	1	
OperationResponse	Response		Out	1	

7.1.5.2.1. Reset Types

The following reset types are supported:

softReset(1): The device performs a soft reset and clears the volatile downloaded configuration. The Amplifier that does not support softReset will perform a hardReset when commanded to do a softReset.

hardReset(2): The device performs a power-on reset and clears the volatile downloaded configuration.

nvReset(3): The device clears most non-volatile configuration and performs a hard reset. Note that nvReset was copied over from [RPHY-OSSI] for consistency with fiber node operations and its applicability to an Amplifier is vendor specific.

factoryReset(4): The device restores the factory configuration and performs a hard reset.

The Amplifier ***shall*** support softReset.

The Amplifier ***shall*** support hardReset.

The amplifier *should* support nvReset.

The Amplifier *may* support factoryReset.

7.1.5.2.1.1. softReset

The softReset provides a partial reset of the Amplifier intended to take steps to hasten the Amplifier initialization process and minimize service interruption.

The softReset resets the Amplifier volatile configuration and operating state, including terminating all connections to servers, releasing IP addresses obtained via DHCP, clearing network authentication information, etc. In addition, Amplifier vendors may choose to reset other Amplifier subsystems to facilitate a more robust recovery from undesirable states. This includes but is not limited to resetting software or hardware subsystems.

In addition, Amplifier vendors can use other methods of hastening the Amplifier initialization during the softReset, such as bypassing steps that it would normally perform during hardReset. This may include, but is not limited to, bypassing reset of hardware subsystems, bypassing diagnostic processes, etc.

The Amplifier ***shall*** perform a softReset whenever commanded to do so, regardless of the Amplifier initialization or operational state.

When performing a softReset, the Amplifier ***shall*** return all volatile configuration and operational state parameters to default values.

The Amplifier ***shall*** retain non-volatile configuration through a softReset.

7.1.5.2.1.2. hardReset

The hardReset is the most comprehensive form of reset that performs a power-on reset and clears all the volatile configuration.

When performing a hardReset, the Amplifier ***shall*** perform a full power cycle, or the equivalent thereof, whereupon the Amplifier returns to a state like the state achieved on initial power up.

The Amplifier ***shall*** retain non-volatile configuration through a hardReset.

7.1.5.2.2. Reset OperationResponse

This parameter is the response to the Reset operation command.

Table 19 – Reset Response Object Parameters

Parameter Name	Type	Required	Type Constraints	Units
Success	Boolean	Yes		
ErrorTag	Enum	No		
ErrorMessage	String	No		

Success
This parameter reports the success or failure of the operation. True indicates the action performed by the operation was successful.

ErrorTag
If the operation was unsuccessful, this parameter reports the Error Tag for the operation. This parameter is not included if the action was successful.

ErrorMessage
If the operation was unsuccessful, this parameter reports the Error Message for the operation. This parameter is not included if the action was successful.

Table 20 – Reset Response Operation Errors

ErrorTag	ErrorMessage
Entity Not Found	This operation does not exist on this device
Not in Valid State	Reset is in progress
Internal Error	The device had an error and could not process the request
Invalid Input	Invalid input parameter
Access denied	The operation request is not authorized
Operation Not Supported	The device does not support the operation or feature

7.1.5.3. DisableAutoReboot Operation

DisableAutoReboot represents a synchronous operation that takes as an input parameter a timeout value. As a successful result of the operation, the auto reboot capability of the Amplifier is delayed until the value in the timeout parameter has elapsed.

Table 21 – DisableAutoReboot Operation Parameters

Parameter Name	Type	Type Constraints	Direction	Multiplicity	Units
Timeout	UnsignedShort		In	1	seconds
OperationResponse	Response		Out	1	

7.1.5.3.1. Timeout

The timeout parameter is several seconds between 1 second and 360 seconds, with a default value of 360 seconds.

Reading this operation always returns the value of Timeout.

7.1.5.3.2. DisableAutoReboot OperationResponse

This parameter is the response to the DisableAutoReboot operation command.

Table 22 – DisableAutoReboot Response Operation Parameters

Parameter Name	Type	Required	Type Constraints	Units
Success	Boolean	Yes		
ErrorTag	Enum	No		
ErrorMessage	String	No		

Success
This parameter reports the success or failure of the operation. True indicates the action performed by the operation was successful.

ErrorTag
If the operation was unsuccessful, this parameter reports the Error Tag for the operation. This parameter is not included if the action was successful.

ErrorMessage
If the operation was unsuccessful, this parameter reports the Error Message for the operation. This parameter is not included if the action was successful.

Table 23 – DisableAutoReboot Response Operation Errors

ErrorTag	ErrorMessage
Entity Not Found	This operation does not exist on this device
Not in Valid State	Reset is in progress
Internal Error	The device had an error and could not process the request
Invalid Input	Invalid input parameter
Access denied	The operation request is not authorized
Operation Not Supported	The device does not support the operation or feature

7.1.5.4. EnableAutoReboot Operation

EnableAutoReboot represents a synchronous operation that has no input parameters. As a successful result of the operation, the auto reboot capability of the Amplifier is enabled, and the value of DisableAutoReboot Timeout is not used.

Reading this operation always returns the value of empty.

Table 24 – EnableAutoReboot Operation Parameters

Parameter Name	Type	Type Constraints	Direction	Multiplicity	Units
OperationResponse	Response		Out	1	

7.1.5.4.1. EnableAutoReboot OperationResponse

This parameter is the response to the EnableAutoReboot operation command.

Table 25 – EnableAutoReboot Response Operation Parameters

Parameter Name	Type	Required	Type Constraints	Units
Success	Boolean	Yes		
ErrorTag	Enum	No		
ErrorMessage	String	No		

Success
This parameter reports the success or failure of the operation. True indicates the action performed by the operation was successful.

ErrorTag
If the operation was unsuccessful, this parameter reports the Error Tag for the operation. This parameter is not included if the action was successful.

ErrorMessage
If the operation was unsuccessful, this parameter reports the Error Message for the operation. This parameter is not included if the action was successful.

Table 26 – EnableAutoReboot Response Operation Errors

ErrorTag	ErrorMessage
Entity Not Found	This operation does not exist on this device
Not in Valid State	Reset is in progress
Internal Error	The device had an error and could not process the request
Invalid Input	Invalid input parameter
Access denied	The operation request is not authorized
Operation Not Supported	The device does not support the operation or feature

7.1.6. Event Status Group (EventStatusGrp)

Event Reporting is described in Section 9.

Figure 15 defines the event status related classes and signals. These are classes have read-only attributes.

Event Status describes a network component event that may be of interest in fault isolation and troubleshooting.

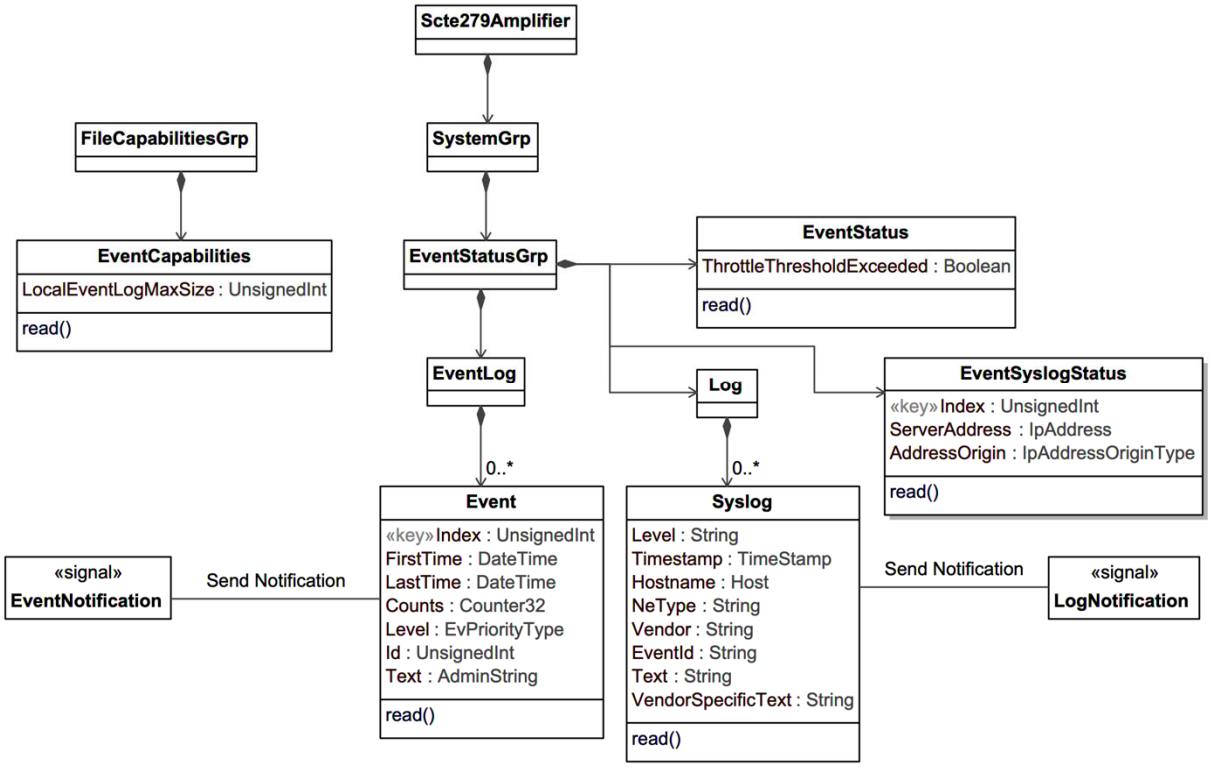

Figure 15 – Event Status Class Diagram

7.1.6.1. EventCapabilities

This attribute describes capabilities of the local nonvolatile log.

Table 27 – EventCapabilities Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
LocalEventLogMaxSize	UnsignedInt	ro			10

7.1.6.1.1. LocalEventLogMaxSize

This attribute describes the maximum number of entries that can be held in the local nonvolatile log.

7.1.6.2. EventStatus

Table 28 – EventStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
ThrottleThresholdExceeded	Boolean	ro			

If true(1), event transmission is currently inhibited due to exceeding the event threshold in the current interval.

If false(0), event transmission in not inhibited.

7.1.6.3. Event

Event describes events.

Table 29 – Event Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Index	UnsignedInt	ro			
FirstTime	DateTime	ro			
LastTime	DateTime	ro			
Counts	Counter32	ro			
Level	EvPriorityType	ro	emergency(1), alert(2), critical(3), error(4), warning(5), notice(6), informational(7), debug(8)		
Id	UnsignedInt	ro			
Text	AdminString	ro			

7.1.6.3.1. Index

This key attribute provides relative ordering of the objects in the event log. This attribute will always increase except when

- the log is reset,
- the device reboots and does not implement non-volatile storage for this log, or
- it reaches the value 2^{31}.

In all the above cases, the next entry for this attribute is 1.

7.1.6.3.2. FirstTime

This attribute provides the local Amplifier time when this event was created.

7.1.6.3.3. LastTime

When an event reports only one event occurrence, this attribute will have the same value as the corresponding instance of FirstTime. When an event reports multiple event occurrences, this attribute will record the local Amplifier time when the most recent occurrence for this event occurred.

7.1.6.3.4. Counts

This attribute provides the number of consecutive event occurrences reported by this event. This starts at 1 with the creation of this event occurrence and increments by 1 for each subsequent duplicate event occurrence.

7.1.6.3.5. Level

This attribute provides the priority level of this event as defined in Section 9. These are ordered from most serious (emergency) to least serious (debug).

7.1.6.3.6. Id

This attribute provides the identifier of this event as defined in Section 9.

7.1.6.3.7. Text

This attribute provides the message text of this event as defined in Section 9.

7.1.6.4. Syslog

Syslog describes Syslog messages.

Table 30 – Syslog Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Level	String	ro			
Timestamp	TimeStamp	ro			
Hostname	Host	ro			
NeType	String	ro	scte279amp		scte279amp
Vendor	String	ro			
EventId	String	ro			
Text	String	ro			
VendorSpecificText	String	ro			

7.1.6.4.1. Level

This attribute is the syslog event priority level as described in Section 9.3.4.3.

7.1.6.4.2. TimeStamp

This attribute is the timestamp associated with the syslog message as described in Section 9.3.4.3.

7.1.6.4.3. Hostname

This attribute is the Hostname associated with the syslog message as described in Section 9.3.4.3.

7.1.6.4.4. NeType

This attribute is the NeType associated with the syslog message. At this time, this field is limited to scte279amp.

7.1.6.4.5. Vendor

This attribute is the vendor associated with the syslog message as described in Section 9.3.4.3.

7.1.6.4.6. EventId

This attribute is the eventId associated with the syslog message as described in Section 9.3.4.3.

7.1.6.4.7. Text

This attribute is the text associated with the syslog message as described in Section 9.3.4.3.

7.1.6.4.8. VendorSpecificText

This attribute is vendor specific text associated with the syslog message as described in Section 9.3.4.3.

7.1.6.5. EventSyslogStatus

Syslog describes Syslog messages.

Table 31 – EventSyslogStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Index	UnsignedInt	ro			
ServerAddress	IpAddress	ro			
AddressOrigin	IpAddressOriginType	ro	other(1), static(2), dhcp(4), linkLayer(5), random(6)		

7.1.6.5.1. Index

This key represents a unique identifier of an instance of this object.

7.1.6.5.2. ServerAddress

This attribute represents the IP address of the Syslog server. If DNS is supported, this attribute can contain the FQDN of the Syslog server.

7.1.6.5.3. AddressOrigin

This attribute represents the origin of the IP address of the Syslog server. The type constraints are borrowed from [RFC 4293].

7.1.7. Event Configuration Group (EventCfgGrp)

Figure 16 shows the Event configuration group of classes which allow the configuration of sending events. These are classes have read-write attributes.

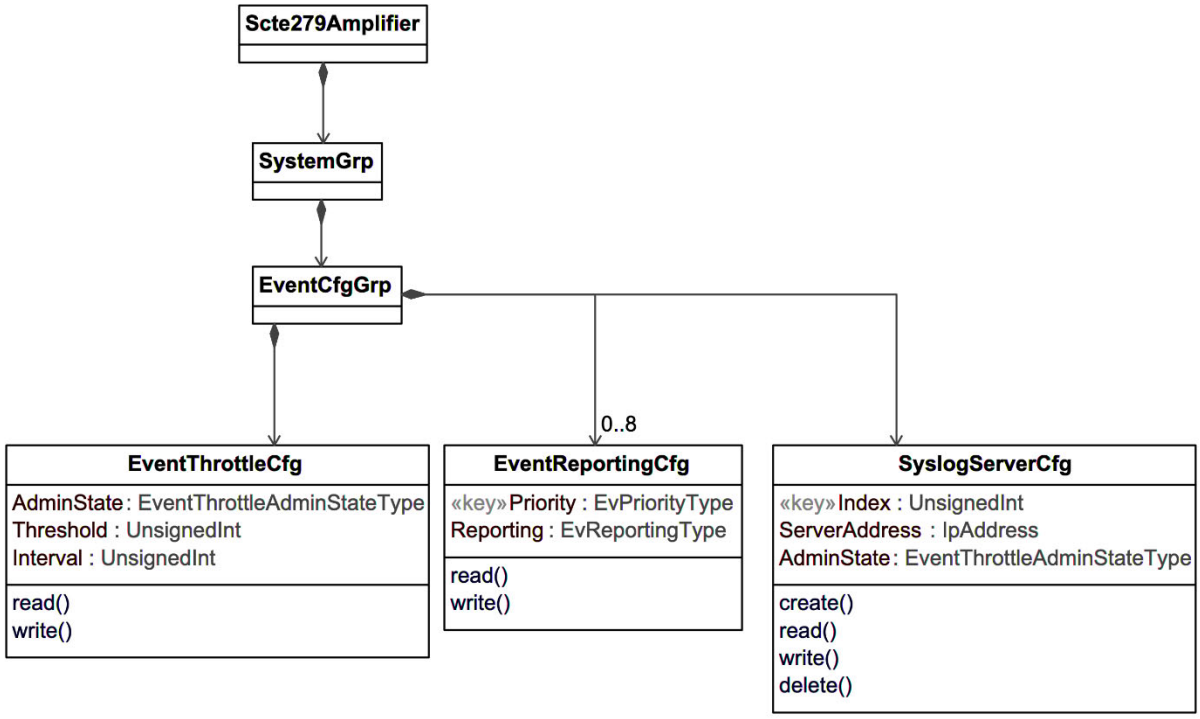

Figure 16 – Event Configuration Class Diagram

7.1.7.1. EventThrottleCfg

This attribute configures the event throttling parameters for the amplifier.

Table 32 – EventThrottleCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
AdminState	EventThrottle AdminStatus Type	rw	unconstrained(1), maintainBelowThreshold(2), stopAtThreshold(3), inhibited(4)		Unconstrained(1)
Threshold	UnsignedInt	rw		events	10
Interval	UnsignedInt	rw		seconds	10

7.1.7.1.1. AdminState

This attribute controls the transmission of traps and syslog messages with respect to the trap pacing threshold. This attribute is defined in [RFC 4639].

unconstrained(1) causes event messages to be transmitted without regard to the threshold settings.

maintainBelowThreshold(2) causes event messages to be suppressed if the number of event messages would otherwise exceed the threshold.

stopAtThreshold(3) causes event message transmission to cease at the threshold, and not resume until directed to do so.

inhibited(4) causes all event messages to be suppressed.

A single event is always treated as a single event for threshold counting. That is, an event causing both a trap and a syslog message is still treated as a single event.

At initial startup, this object has a default value of unconstrained(1). Writing to this object resets the thresholding state.

7.1.7.1.2. Threshold

This attribute configures the number of Syslog notifications per interval permitted before throttling is applied.

A single event is always treated as a single event for threshold counting. That is, an event causing both a trap and a syslog message is still treated as a single event.

7.1.7.1.3. Interval

This attribute configures the interval over which the throttling applies.

7.1.7.2. EventReportingCfg

This attribute configures the event reporting parameters for the amplifier.

Table 33 – EventReportingCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Priority	EvPriorityType	rw	emergency(1), alert(2), critical(3), error(4), warning(5), notice(6), informational(7), debug(8)		
Reporting	EvReportingType	rw	Bit 1 traps(1), Bit 2 syslog(2), Bit 8 localVolatile(8), Bit 9 stdInterface(9)		

7.1.7.2.1. Priority

This attribute is the priority of the event. It matches a priority defined in Section 9.

7.1.7.2.2. Reporting

This attribute defines the ways that the Amplifier will report the event, using information from [RFC 4639]
Permissible BIT values for [RFC 4639] docsDevEvReporting objects include:

local(0)
traps(1)
syslog(2)
localVolatile(8)
stdInterface(9)

Bit-0 means non-volatile Local Log storage and bit-8 is used for volatile Local Log storage (see Section 9.3.1).

Bit-1 means SNMP Notifications which correspond to both SNMP Trap and SNMP Inform.

Bit-2 means Syslog.

Bit-8 means localVolatile

Bit-9 means YANG Notifications

Since these are on an octet boundary, the Amplifier ***shall*** ignore bits 3 through 7 as possible BIT values for docsDevEvReporting.

Implementations may not support all options for all event classes and at a minimum should allow traps and syslog to be disabled.

If the local(0) bit is set, then log to the internal log.

If the traps(1) bit is set, then generate an SNMP trap.

If the syslog(2) bit is set, then send a syslog message (assuming a syslog address is set).

If the localVolatile(8) bit is set, then log to the internal log without updating non-volatile store.

If the stdInterface(9) bit is set, then the agent ignores all other bits except the local(0), syslog(2), and localVolatile(8) bits. Setting the stdInterface(9) bit indicates that [RFC 3413] and [RFC 3014] are being used to control event reporting mechanisms.

7.1.7.3. SyslogServerCfg

These attributes control the transmission of syslog messages. Syslog server(s) are generally made known during the DHCP configuration of the amplifier.

Table 34 – SyslogServerCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Index	UnsignedInt	crwd			
ServerAddress	Host	crwd			
AdminState	EventThrottleAdminStateType	crwd	unconstrained(1), maintainBelowThreshold(2), stopAtThreshold(3), inhibited(4)		Unconstrained(1)

7.1.7.3.1. Index

The index configures messaging to a specific syslog server.

7.1.7.3.2. ServerAddress

This is the IP address of the syslog server.

7.1.7.3.3. AdminStatus

This attribute controls the transmission of syslog messages with respect to the trap pacing threshold.

unconstrained(1) causes syslog messages to be transmitted without regard to the threshold settings.

maintainBelowThreshold(2) causes syslog messages to be suppressed if the number of event messages would otherwise exceed the threshold.

stopAtThreshold(3) causes syslog message transmission to cease at the threshold, and not resume until directed to do so.

inhibited(4) causes all trap transmission and syslog messages to be suppressed.

A single event is always treated as a single event for threshold counting. That is, an event causing both a trap and a syslog message is still treated as a single event.

At initial startup, this object has a default value of unconstrained(1). Writing to this object resets the thresholding state.

7.1.8. Event Control Component Diagram

Figure 17 shows the EventControl component diagram and illustrates the Amplifier and Network Management application components that provide event control operations on the Amplifier.

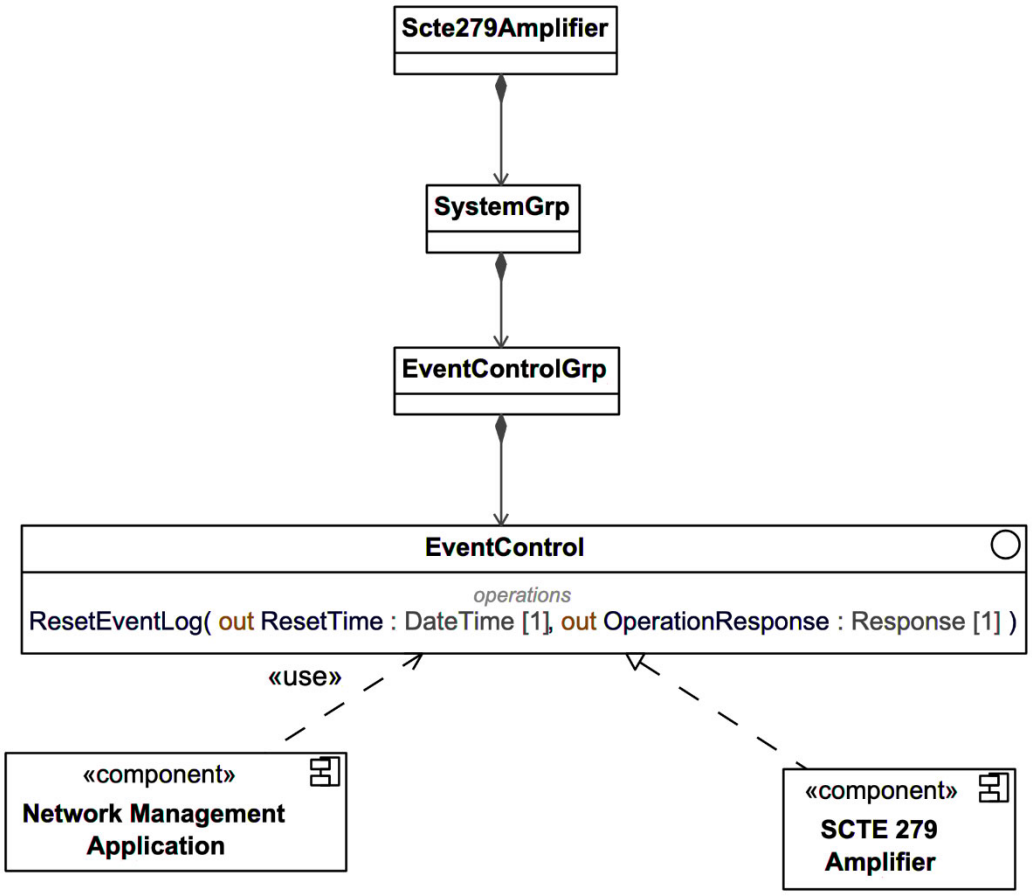

Figure 17 – Event Control Component Diagram

7.1.8.1. EventControl

The ResetEventLog interface manages resetting the event log.

Table 35 – ResetEventLog Operation Parameters

Parameter Name	Type	Type Constraints	Direction	Multiplicity	Units
ResetTime	DateTime		Out	1	
OperationResponse	Response		Out	1	

7.1.8.1.1. ResetEventLog Operation

ResetEventLog represents a synchronous operation that has no input parameters. As a successful result of the operation, the Amplifier event log is cleared.

Reading this operation always returns the value of empty.

7.1.8.1.2. ResetEventLog OperationResponse

This parameter is the response to the ResetEventLog operation command.

Table 36 – ResetEventLog Response Parameters

Parameter Name	Type	Required	Type Constraints	Units
Success	Boolean	Yes		
ErrorTag	Enum	No		
ErrorMessage	String	No		

Success
This attribute reports the success or failure of the operation. True indicates the action performed by the operation was successful.

ErrorTag
If the operation was unsuccessful, this parameter reports the Error Tag for the operation. This parameter is not included if the action was successful.

ErrorMessage
If the operation was unsuccessful, this parameter reports the Error Message for the operation. This parameter is not included if the action was successful.

Table 37 – ResetEventLog Response Operation Errors

ErrorTag	ErrorMessage
Entity Not Found	This operation does not exist on this device
Not in Valid State	Reset is in progress
Internal Error	The device had an error and could not process the request
Invalid Input	Invalid input parameter
Access denied	The operation request is not authorized
Operation Not Supported	The device does not support the operation or feature

7.1.9. File Management

Figure 18 shows the File Management group of classes which enable the management of the supported file types. File Management is rooted under the System Group.

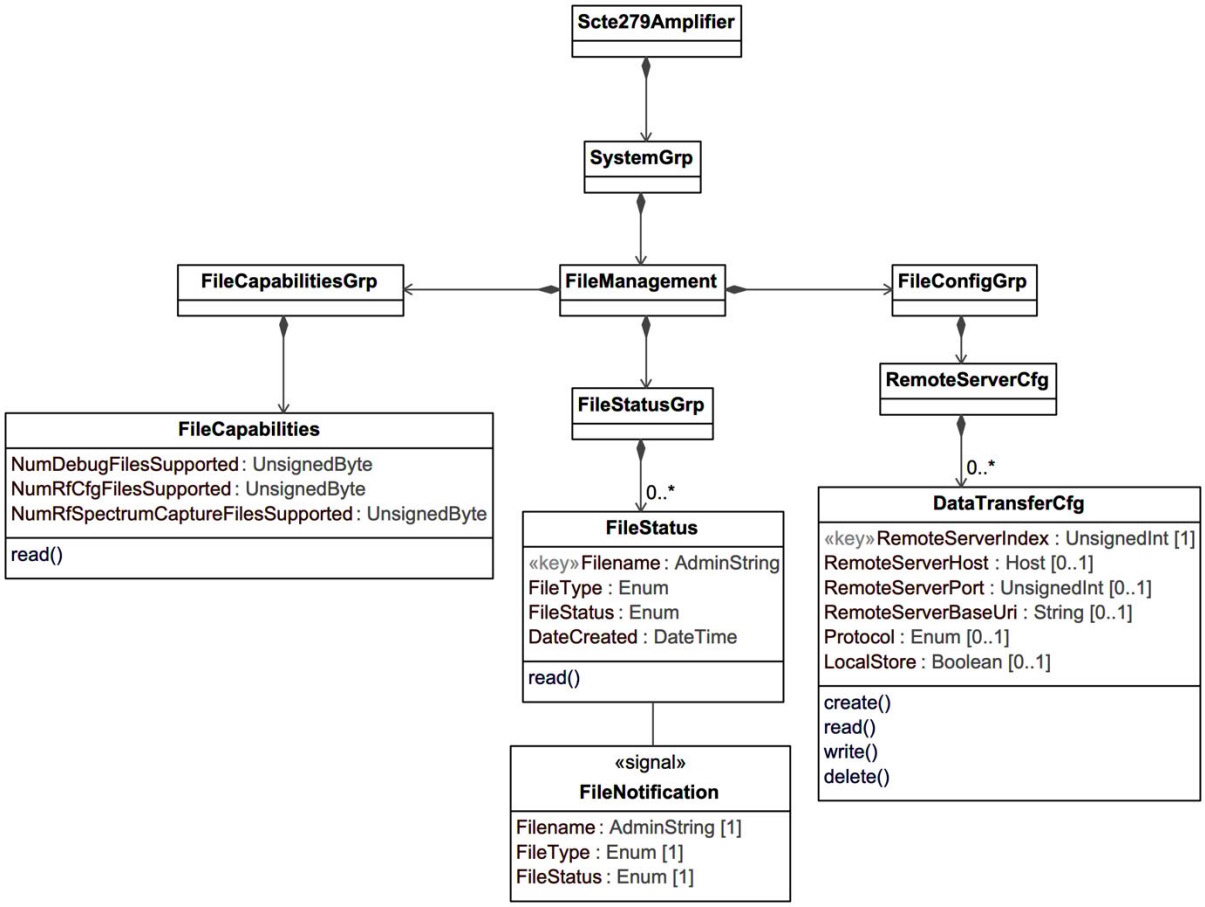

Figure 18 – File Management Class Diagram

7.1.9.1. FileCapabilities

The eAMP ***shall*** be able to store data files (inclusive) that contain at least 64 kilobytes of data.

If the eAMP is commanded to collect and store data that exceeds its file size capability, the eAMP *may* stop collecting and storing data and report an error to the application that commanded it to collect data and store such data.

Table 38 – FileCapabilities Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
NumDebugFilesSupported	UnsignedByte	ro		files	
NumRfCfgFilesSupported	UnsignedByte	ro		files	
NumRfSpectrumCaptureFilesSupported	UnsignedByte	ro		files	

7.1.9.1.1. NumDebugFilesSupported

This attribute identifies how many Debug files are supported by the Amplifier.

7.1.9.1.2. NumRfCfgFilesSupported

This attribute identifies how many detailed RF configuration files are supported by the Amplifier.

7.1.9.1.3. NumRfSpectrumCaptureFilesSupported

This attribute identifies how many RF spectrum capture files are supported by the Amplifier.

7.1.9.2. FileStatus

The FileStatus class provides the status attributes for each data file stored locally on the Amplifier. An instance is created for each file that is available, in the target Amplifier, for upload.

The Amplifier could have limited resources to store data files; therefore, if the number of files exceeds the minimum supported number of files requirements for the device, newly created files can overwrite/replace existing files as new data files become available.

The Amplifier ***shall*** create an instance of FileStatus for each data file that is available for upload.

If a data file is no longer available for upload, the Amplifier ***shall*** remove that file's instance from the FileStatus object.

Table 39 – FileStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Filename	AdminString	ro			
FileType	Enum	ro	other(1), debug(2), rfCfg(3), rfSpectrumCapture(4)		
FileStatus	Enum	ro	other(1), availableForUpload(2), uploadInProgress(3), uploadCompleted(4), uploadPending(5), uploadCancelled(6) error(7)		
DateCreated	DateTime	ro			

7.1.9.2.1. Filename

This attribute contains the name of the data file stored on the device, that is available to be uploaded to the File Server. Filenames are defined by the application that creates them.

7.1.9.2.2. FileType

This attribute reports the type of the data file. The possible values are listed below.

- other(1) - Any file type not covered by the other defined values

- debug(2) – a debug file
- rfCfg(3) – a detailed RF configuration file
- rfSpectrumCapture(4) – an RF spectrum capture file

A debug file contains information useful for postmortem analysis of an error that occurs on the Amplifier. The format of the debug file is vendor specific. There will be an instance of this object for each file that is available in the device for upload or for which an upload is in progress. The Amplifier could have limited resources to save captured debug files; therefore, if the number of generated files exceeds the supported number for the device, newly-created instances can overwrite/replace existing instances as new debug files become available.

An rfCfg file provides detailed information on the RF configuration of the amplifier. The format of the rfCfg file is vendor specific.

This SpectrumCapture file is associated with the eAMP and is further described in Section 7.4.3.

7.1.9.2.3. FileStatus

This attribute reports the status of the data file. The possible values are listed below.

- other(1) - Any condition not covered by the other defined values.
- availableForUpload(2) - The file is available to be uploaded.
- uploadInProgress(3) - The file is currently being uploaded.
- uploadCompleted(4) - The file was successfully uploaded.
- uploadPending(5) - The file has been selected for upload, but a condition does not allow the upload to take place. The upload will start when the condition blocking uploads has been removed. For example, another upload that is currently in progress could cause this value to be returned.
- uploadCancelled(6) - An upload was cancelled before it completed.
- error(7) - An error occurred and the file was not successfully uploaded.

7.1.9.2.4. DateCreated

This attribute reports the date and time of when the local file was created on the Network Function's local file system.

7.1.9.3. FileNotification

FileNotification is an asynchronous notification informing the File Server about the status of data file. This notification is sent by the Amplifier when a data file is created and ready to upload. It can also be sent when the status of a file is updated by the Amplifier (e.g., when an upload is cancelled).

When the status of a locally stored file changes, the Amplifier ***shall*** log Event ID 70000704.

Table 40 – FileNotification Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Filename	AdminString	ro			
FileType	Enum	ro	other(1), debug(2), rfCfg(3), rfSpecCapture(4)		
FileStatus	Enum	ro	other(1), availableForUpload(2), uploadInProgress(3), uploadCompleted(4), uploadPending(5), uploadCancelled(6), error(7)		

7.1.9.3.1. Filename

This attribute contains the name of the data file stored on the device, that is available to be uploaded to the File Server. Filenames are defined by the application that creates them.

7.1.9.3.2. FileType

This attribute reports the type of the data file. See Section 7.1.9.2.2.

7.1.9.3.3. FileStatus

This attribute reports the status of the data file. See Section 7.1.9.2.3.

7.1.9.4. DataTransferCfg

The DataTransferCfg class defines the configuration of destinations for files, such as debug files.

The Amplifier ***shall*** support creation and deletion of multiple instances of the DataTransferCfg object.

Table 41 – DataTransferCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
RemoteServerIndex	UnsignedInt	crwd			
RemoteServerHost	Host	crwd			
RemoteServerPort	UnsignedInt	crwd			
RemotServerBaseUri	String	crwd			
Protocol	Enum	crwd	other(1), tftp(2)		
LocalStore	Boolean	crwd			

7.1.9.4.1. RemoteServerIndex

This key attribute uniquely identifies a destination for a file.

7.1.9.4.2. RemoteServerHost

This attribute identifies a destination host address for a file.

7.1.9.4.3. RemoteServerPort

This attribute identifies a destination port number for a file. If the value of LocalStore is 'false', then RemoteServerHost, RemoteServerUri, and Protocol are required attributes.

7.1.9.4.4. RemoteServerBaseUri

This attribute identifies a destination base Uniform Resource Identifier for a file. This attribute does not contain the actual filename. If the value of LocalStore is 'false', then RemoteServerHost, RemoteServerBaseUri, and Protocol are required attributes.

7.1.9.4.5. Protocol

This attribute identifies the data transfer protocol for a file.

other(1) indicates the Amplifier will use some other protocol to transfer the data.
tftp(2) indicates the Amplifier will use TFTP to transfer the file.

7.1.9.4.6. LocalStore

This attribute identifies whether the device stores files locally. If the value is set to ‘true’, the device will store the file locally on the device. If the value is set to ‘false’, the device will not store the file locally on the device and will rely on the other attributes for where to send the file. If the value of LocalStore is 'false', then RemoteServerHost, RemoteServerUri, and Protocol are required attributes.

7.1.10. File Management Component Diagram

Figure 19 shows the FileManagement component diagram and illustrates the Amplifier and Network Management application components that provide file management operations on the Amplifier.

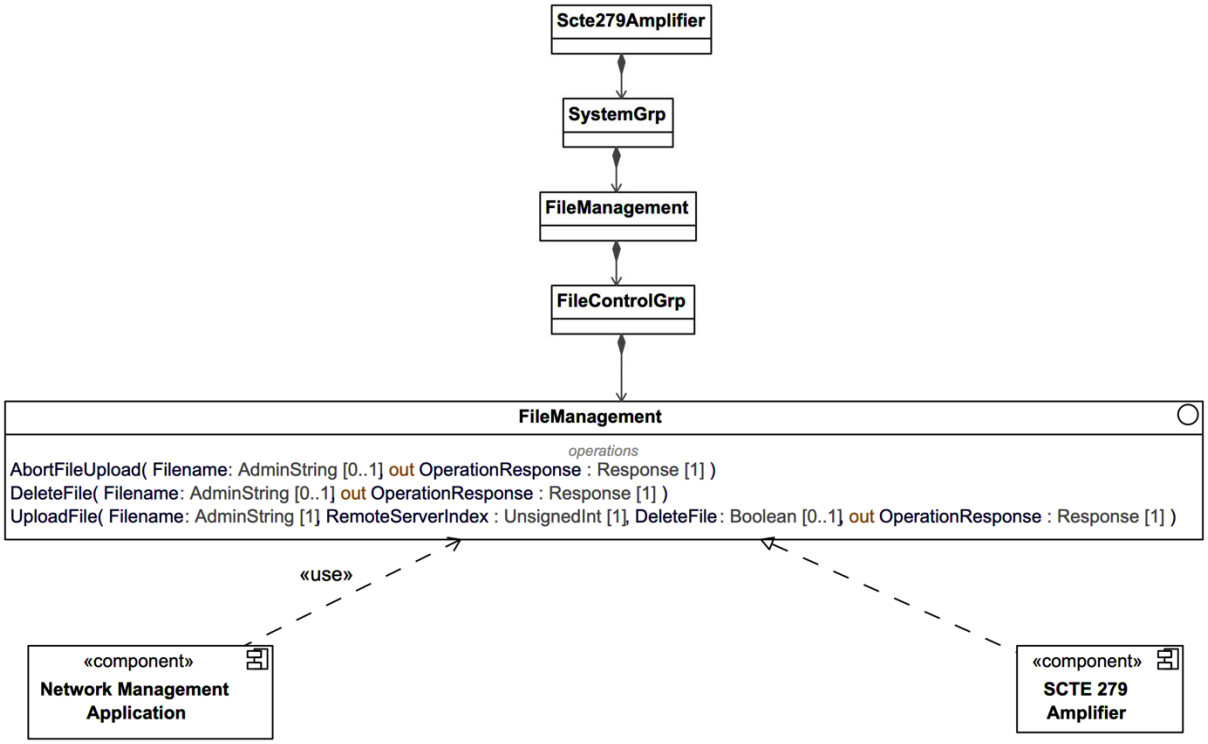

Figure 19 – File Management Component Diagram

The FileManagement interface manages file operations associated with the Amplifier. Operations include:

- AbortFileUpload
- DeleteFile
- UploadFile

7.1.10.1. AbortFileUpload Operation

AbortFileUpload is a synchronous operation that aborts, or cancels, a pending file upload or an upload currently in progress. If the Filename parameter is not provided, all pending or currently uploading files will be aborted.

A successful operation will result in the Amplifier aborting a pending file upload or a file upload currently in progress. The FileStatus for the file identified by the Filename will be changed to 'uploadCancelled'.

Table 42 – AbortFileUpload Operation Parameters

Parameter Name	Type	Type Constraints	Direction	Multiplicity	Units
Filename	AdminString		In	0..1	
OperationResponse	Response		Out	1	

7.1.10.1.1. AbortFileUpload Parameter Definitions

7.1.10.1.1.1. Filename

The Filename parameter contains the name of the file stored on the device.

7.1.10.1.1.2. OperationResponse

This parameter is the response to the AbortFileUpload operation command.

Table 43 – AbortFileUpload Response Object Attributes

Parameter Name	Type	Required	Type Constraints	Units
Success	Boolean	Yes		
ErrorTag	Enum	No		
ErrorMessage	String	No		

Success
This parameter reports the success or failure of the operation. True indicates the action performed by the operation was successful.

ErrorTag
If the operation was unsuccessful, this parameter reports the Error Tag for the operation. This parameter is not included if the action was successful.

ErrorMessage
If the operation was unsuccessful, this parameter reports the Error Message for the operation. This parameter is not included if the action was successful.

Table 44 – AbortFileUpload Response Operation Errors

ErrorTag	ErrorMessage
Entity Not Found	Filename does not exist on this device
Not in Valid State	File upload not in progress
Internal Error	The device had an error and could not process the request
Invalid Input	Invalid input parameter
Access denied	The operation request is not authorized
Operation Not Supported	The device does not support the operation or feature

7.1.10.2. DeleteFile Operation

DeleteFile is a synchronous operation that deletes a file stored locally on the target device. If the Filename parameter is not provided, all locally stored files will be deleted.

A successful operation will result in the Amplifier deleting a stored file from local memory. The FileStatus instance of the file identified by the Filename will be deleted. An unsuccessful operation will occur if a file upload is currently in progress.

Table 45 – DeleteFile Operation Parameters

Parameter Name	Type	Type Constraints	Direction	Multiplicity	Units
Filename	AdminString		In	0..1	
OperationResponse	Response		Out	1	

7.1.10.2.1. DeleteFile Parameter Definitions

7.1.10.2.1.1. Filename

The Filename parameter contains the name of the file stored on the device.

7.1.10.2.1.2. OperationResponse

This parameter is the response to the DeleteFile operation command.

Table 46 – DeleteFile Response Object Attributes

Parameter Name	Type	Required	Type Constraints	Units
Success	Boolean	Yes		
ErrorTag	Enum	No		
ErrorMessage	String	No		

Success
This parameter reports the success or failure of the operation. True indicates the action performed by the operation was successful.

ErrorTag
If the operation was unsuccessful, this parameter reports the Error Tag for the operation. This parameter is not included if the action was successful.

ErrorMessage
If the operation was unsuccessful, this parameter reports the Error Message for the operation. This parameter is not included if the action was successful.

Table 47 – DeleteFile Response Operation Errors

ErrorTag	ErrorMessage
Entity Not Found	Filename does not exist on this device
Not in Valid State	Reset is in progress
Internal Error	The device had an error and could not process the request
Invalid Input	Invalid input parameter
Access denied	The operation request is not authorized
Operation Not Supported	The device does not support the operation or feature

7.1.10.3. UploadFile Operation

UploadFile is a non-blocking asynchronous operation that uploads a file stored locally on the Amplifier using an instance of DataTransferCgf.

The file specified by Filename from the FileStatus class, with a FileStatus of 'availableForUpload', will be uploaded to the configured destination server specified by RemoteServerIndex of the DataTransferCfg class. If the DeleteFile parameter is not provided, the file will be retained in local storage.

A successful operation will result in the Amplifier initiating a file upload using the configured file transfer method. The FileStatus instance of the file identified by the Filename will be updated to 'uploadInProgress'. If the Amplifier is unable to immediately initiate the file transfer due to another blocking condition, the FileStatus instance of the file identified by the Filename will be updated to 'uploadPending'. The upload will be initiated when the blocking condition has been removed, for example, another upload in progress.

An unsuccessful operation will occur if the specified file, or any file upload, is currently in progress. For any unsuccessful operation, the FileStatus instance of the file identified by the Filename will be updated to 'error'.

Table 48 – UploadFile Operation Parameters

Parameter Name	Type	Type Constraints	Direction	Multiplicity	Units
Filename	AdminString		In	1	
RemoteServerIndex	UnsignedInt		In	1	
DeleteFile	Boolean		In	0..1	
OperationResponse	Response		Out	1	

7.1.10.3.1. UploadFile Parameter Definitions

7.1.10.3.1.1. Filename

The Filename parameter contains the name of the file stored on the device.

7.1.10.3.1.2. RemoteServerIndex

This parameter specifies the remote server defined in an instance of the DataTransferCfg class.

7.1.10.3.1.3. DeleteFile

This parameter is a Boolean flag indicating whether the Amplifier is to retain the file in local storage. If set to ‘true’, the target device will delete the locally stored file after successful upload. If set to ‘false’, the target device will retain the locally stored file after a successful upload.

7.1.10.3.1.4. OperationResponse

This parameter is the response to the UploadFile command.

Table 49 – UploadFile Response Object Attributes

Parameter Name	Type	Required	Type Constraints	Units
Success	Boolean	Yes		
ErrorTag	Enum	No		
ErrorMessage	String	No		

Success
This parameter reports the success or failure of the operation. True indicates the action performed by the operation was successful.

ErrorTag
If the operation was unsuccessful, this parameter reports the Error Tag for the operation. This parameter is not included if the action was successful.

ErrorMessage
If the operation was unsuccessful, this parameter reports the Error Message for the operation. This parameter is not included if the action was successful.

Table 50 – UploadFile Response Operation Errors

ErrorTag	ErrorMessage
Entity Not Found	Filename does not exist on this device
Not in Valid State	Reset is in progress
Internal Error	The device had an error and could not process the request
Invalid Input	Invalid input parameter
Access denied	The operation request is not authorized
Operation Not Supported	The device does not support the operation or feature

7.2. RF Group (RfGrp)

Figure 20 shows the Rf Group of classes which define the RF capabilities and RF status of the Amplifier and the ports on the Amplifier. These are classes have read-only attributes.

The Amplifier ***shall*** implement all classes described in the RfGrp information model.

7.2.1. RF Capabilities Group (RfCapabilitiesGrp)

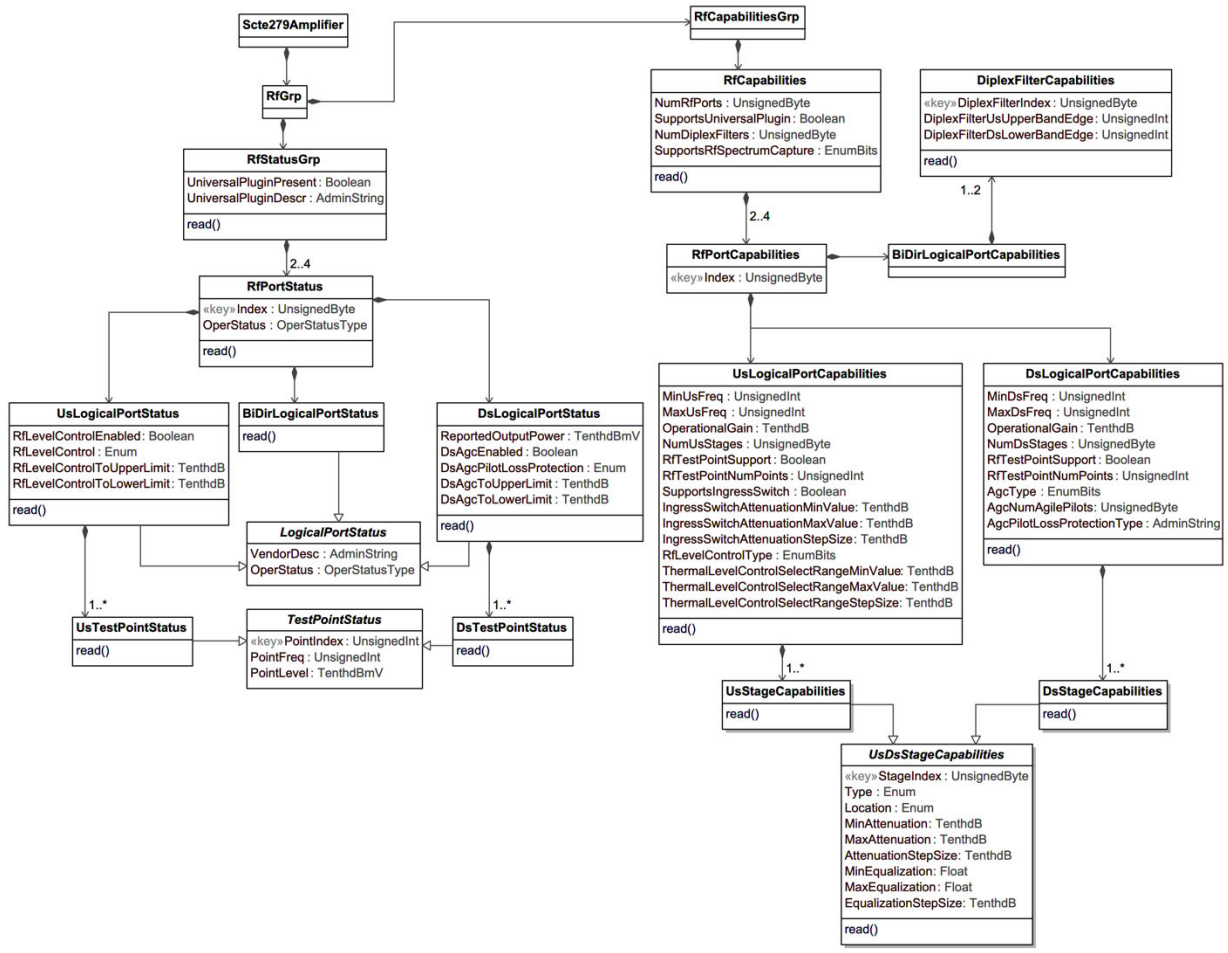

Figure 20 – RF Group Status Class Diagram

7.2.1.1. RF Capabilities

This class describes the RF capabilities of the Amplifier.

Table 51 – RfCapabilities Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
NumRfPorts	UnsignedByte	ro			
SupportsUniversalPlugin	Boolean	ro			
NumDiplexFilters	UnsignedByte	ro			
SupportsRfSpectrumCapture	EnumBits	ro	notSupported(0) port1Upstream(1) port1Downstream(2) port2Upstream(3) port2Downstream(4) port3Upstream(5) port3Downstream(6) port4Upstream(7) port4Downstream(8)		

7.2.1.1.1. NumRfPorts

Reports the number of RF ports on the Amplifier that can support active RF signals. See Figure 7 for additional information.

7.2.1.1.2. SupportsUniversalPlugin

Reports if the universal plugin is supported. See [SP 920] for additional information on the Universal Plug-in.

7.2.1.1.3. NumDiplexFilters

Reports the number of diplex filters installed. There is an assumption that each physical port will have the same number of diplex filters installed which can be switched into use in the amplifier. The diplex filter is applied per port.

Because a diplex filter affects both the upstream and downstream frequency ranges, Figure 20 references a constraint on the association between RfPortCapabilities and DiplexFilterCapabilities to indicate only BiDirRfPorts have DiplexFilterCapabilities.

Generally, each physical port on the Amplifier will use the same diplex filter. While there might be cases where different physical ports use different diplex filters, there is an assumption that each physical port will have the same diplex filter in operation.

7.2.1.1.4. SupportsRfSpectrumCapture

Reports if the eAMP supports the RF spectrum capture feature and on which ports and the upstream or downstream.

7.2.1.2. RF Port Capabilities

This class describes the RF port capabilities of the Amplifier.

Table 52 – RfPortCapabilities Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Index	UnsignedByte	ro			

7.2.1.2.1. Index

The port index associated with the physical RF port capabilities.

7.2.1.3. Diplex Filter Capabilities

This class describes the diplex filter(s) available on this physical port. Up to two diplex filters are available per port. If two diplex filters are supported, the active diplex filter can be selected in the RF configuration as described in Section 7.2.3.2.

A diplex filter is characterized by its lower band edge (which is the upstream upper band edge) and its upper band edge (which is the downstream lower band edge).

Table 53 – DiplexFilterCapabilities Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
DiplexFilterIndex	UnsignedByte	ro			
DiplexFilterUsUpperBandEdge	UnsignedInt	ro		MHz	
DiplexFilterDsLowerBandEdge	UnsignedInt	ro		MHz	

7.2.1.3.1. DiplexFilterIndex

The index used to get to identify an available diplex filter.

7.2.1.3.2. DiplexFilterUsUpperBandEdge

The upper band edge of the upstream for the diplex filter.

7.2.1.3.3. DiplexFilterDsLowerBandEdge

The lower band edge of the downstream for the diplex filter.

7.2.1.4. Upstream Logical Port Capabilities

This class describes the upstream RF capabilities available on the upstream logical port.

Table 54 – UsLogicalPortCapabilities Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
MinUsFreq	UnsignedInt	ro		MHz	
MaxUsFreq	UnsignedInt	ro		MHz	
Operational Gain	TenthdB	ro			
NumUsStages	UnsignedByte	ro	starts at 1		
RfTestPointSupport	Boolean	ro			
RfTestPointNumPoints	UnsignedInt	ro	starts at 1		
SupportsIngressSwitch	Boolean	ro			
IngressSwitchAttenuationMinValue	TenthdB	ro			
IngressSwitchAttenuationMaxValue	TenthdB	ro			
IngressSwitchAttenuationStepSize	TenthdB	ro			
RfLevelControlType	EnumBits	ro	other(1), unknown(2), quasiAgc(3), thermalLevelControl(4), thermalLevelControlSelect(5)		
ThermalLevelControlSelectRangeMinValue	TenthdB	ro			
ThermalLevelControlSelectRangeMaxValue	TenthdB	ro			
ThermalLevelControlSelectRangeStepSize	TenthdB	ro			

7.2.1.4.1. MinUsFreq

The lowest upstream frequency supported.

7.2.1.4.2. MaxUsFreq

The highest upstream frequency supported. This value depends on the active diplex filter.

7.2.1.4.3. Operational Gain

The upstream operational gain for this upstream logical port.

7.2.1.4.4. NumUsStages

The number of configurable stages for this upstream logical port and starts with stage 1. A stage is either an attenuation or an equalization stage.

7.2.1.4.5. RfTestPointSupport

Indicates if electronically reading the upstream RF test point is supported.

7.2.1.4.6. RfTestPointNumPoints

If RfTestPointSupport is True, then this object indicates the number of test points that can be read for the upstream and starts with test point 1.

If RfTestPointSupport is False, this object is undefined.

7.2.1.4.7. SupportsIngressSwitch

Indicates if the amplifier port supports an upstream ingress switch. Please refer to Figure 7 as this function is usually on an upstream input port.

7.2.1.4.8. IngressSwitchAttenuationMinValue

Indicates the lowest configurable attenuation on the upstream ingress switch for this port.

7.2.1.4.9. IngressSwitchAttenuationMaxValue

Indicates the highest configurable attenuation on the upstream ingress switch for this port.

7.2.1.4.10. IngressSwitchAttenuationStepSize

Indicates the step size for the upstream ingress switch for this port.

7.2.1.4.11. RfLevelControlType

Indicates the upstream RF level control types available on this port. [SP 920] identifies methods up upstream level control.

7.2.1.4.12. ThermalLevelControlSelectRangeMinValue

If upstream thermal level control is supported, this object indicates the lowest configurable upstream thermal level control attenuation for this port.

If upstream thermal level control is not supported, this object is undefined.

7.2.1.4.13. ThermalLevelControlSelectRangeMaxValue

If upstream thermal level control is supported, this object indicates the highest configurable upstream thermal level control attenuation for this port.

If upstream thermal level control is not supported, this object is undefined.

7.2.1.4.14. ThermalLevelControlSelectRangeStepSize

If upstream thermal level control is supported, this object indicates the step size for upstream thermal level control attenuation for this port.

If upstream thermal level control is not supported, this object is undefined.

7.2.1.5. Downstream Logical Port Capabilities

This class describes the RF capabilities available on the downstream logical port.

Table 55 – DsLogicalPortCapabilities Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
MinDsFreq	UnsignedInt	ro		MHz	
MaxDsFreq	UnsignedInt	ro		MHz	
OperationalGain	TenthdB	ro			
NumDsStages	UnsignedByte	ro	starts at 1		
RfTestPointSupport	Boolean	ro			
RfTestPointNumPoints	UnsignedInt	ro	starts at 1		
AgcType	EnumBits	ro	other(1), unknown(2), mfAgc(3), agileAgc(4), agcPilotLossProtection(5)		
AgcNumAgilePilots	UnsignedByte	ro			
AgcPilotLossProtectionType	AdminString	ro			

7.2.1.5.1. MinDsFreq

The lowest downstream frequency supported. This value depends on the active diplex filter.

7.2.1.5.2. MaxDsFreq

The highest downstream frequency supported.

7.2.1.5.3. Operational Gain

The downstream operational gain for this downstream logical port.

7.2.1.5.4. NumDsStages

The number of configurable stages for this downstream logical port and starts with stage number 1. A stage is either an attenuation or an equalization stage.

7.2.1.5.5. RfTestPointSupport

Indicates if electronically reading the downstream RF test point is supported.

7.2.1.5.6. RfTestPointNumPoints

If RfTestPointSupport is True, then this object indicates the number of test points that can be read for the downstream, starting with test point 1.

If RfTestPointSupport is False, this object is undefined.

7.2.1.5.7. AgcType

Indicates the types of AGC that are available on this downstream logical port. The following selections are discussed in [SCTE 279].

other(1)
unknown(2)
mfAgc(3) is multi-frequency AGC.
agileAgc(4) indicates the AGC pilots are agile.
agcPilotLossProtection(5) indicates that a method of AGC pilot loss protection is implemented.

7.2.1.5.8. AgcNumAgilePilots

Indicates the number of agile pilots supported on this port.

7.2.1.5.9. AgcPilotLossProtectionType

Indicates the types of pilot loss protection available on this port. This entry is vendor specific and is a description of the type of pilot loss protection.

7.2.1.6. UsDsStageCapabilities

This class describes the RF capabilities available on either the logical upstream or logical downstream port.

Table 56 – UsDsStageCapabilities Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
StageIndex	UnsignedByte	ro			
Type	Enum	ro	other(1), unknown(2), attenuation(3), equalization(4)		
Location	Enum	ro	other(1), unknown(2), inputStage(3), interStage(4), ouputStage(5)		
MinAttenuation	TenthdB	ro			
MaxAttenuation	TenthdB	ro			
AttenuationStepSize	TenthdB	ro			
MinEqualization	Byte	ro			
MaxEqualization	Byte	ro			
EqualizationStepSize	TenthdB	ro			

7.2.1.6.1. StageIndex

This attribute indicates the number of the stage index. This value starts at 1 and is bounded by either NumUsStages or NumDsStages.

7.2.1.6.2. Type

Indicates the type of stage, either attenuation or equalization.

7.2.1.6.3. Location

Indicates the location of the stage. The definition is vendor specific.

7.2.1.6.4. MinAttenuation

If this is an Attenuation stage, this object indicates the lowest configurable electronically controlled attenuation setting for this stage on this port.

If this is an equalization stage, this object is undefined.

7.2.1.6.5. MaxAttenuation

If this is an Attenuation stage, this object indicates the highest configurable electronically controlled attenuation setting for this stage on this port.

If this is an equalization stage, this object is undefined.

7.2.1.6.6. AttenuationStepSize

If this is an Attenuation stage, this object indicates the electronically controlled attenuation step size for this stage on this port.

If this is an equalization stage, this object is undefined.

7.2.1.6.7. MinEqualization

If this is an Equalization stage, this object indicates the lowest configurable electronically controlled equalization setting for this stage on this port. This value can be a negative number.

If this is an attenuation stage, this object is undefined.

7.2.1.6.8. MaxEqualization

If this is an Equalization stage, this object indicates the highest configurable electronically controlled equalization setting for this stage on this port.

If this is an attenuation stage, this object is undefined.

7.2.1.6.9. EqualizationStepSize

If this is an Equalization stage, this object indicates the downstream electronically controlled equalization step size for this stage on this port.

If this is an attenuation stage, this object is undefined.

7.2.2. RF Status Group (RfStatusGrp)

Reports the Status of the RF ports on the amplifier.

Table 57 – RfStatusGrp Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
UniversalPluginPresent	Boolean	ro			
UniversalPluginDescr	AdminString	ro			notPopulated

7.2.2.1. UniversalPluginPresent

Reports if the universal plugin is present. True means the universal plugin is present. False means the universal plugin is not present.

7.2.2.2. UniversalPluginDescr

If the Universal Plug-in is populated, this attribute reports a description used by the vendor to identify the Universal Plug-in. The format is vendor specific.

If the Universal Plug-in is not populated, this attribute reports notPopulated.

7.2.2.3. UsLogicalPortStatus

Reports the RF Port status for ports on the amplifier.

Table 58 – UsLogicalPortStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
RfLevelControlEnabled	Boolean	ro			
RfLevelControl	Enum	ro	other(1), unknown(2), quasiAgc(3), thermalLevelControl(4)		
RfLevelControlToUpperLimit	TenthdB	ro			
RfLevelControlToLowerLimit	TenthdB	ro			

7.2.2.3.1. RfLevelControlEnabled

An indication of RF level control enabled on the upstream port. True means enabled. False means not enabled.

7.2.2.3.2. RfLevelControl

Reports the method of upstream RF level control in use on this port.

7.2.2.3.3. RfLevelControlToUpperLimit

An indication of the amount of RF level control available before the value hits the upper limit of RF level control.

7.2.2.3.4. RfLevelControlToLowerLimit

An indication of the amount of RF level control available before the value hits the lower limit of RF level control.

7.2.2.4. DsLogicalPortStatus

Reports the RF Port status for ports on the amplifier.

Table 59 – DsLogicalPortStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
ReportedOutputPower	TenthdBmV	ro		dBmV	
DsAgcEnabled	Boolean	ro			
DsAgcPilotLossProtection	Enum	ro	other(1), unknown(2), alternatePilotFreq(3), thermal(4), lockSettings(5)		
DsAgcToUpperLimit	TenthdB	ro			
DsAgcToLowerLimit	TenthdB	ro			

7.2.2.4.1. ReportedOutputPower

Reports the output total composite power for the total occupied spectrum for the selected downstream logical port. The units are dBmV.

Devices with unknown downstream output power report a value of 0.

7.2.2.4.2. DsAgcEnabled

Indicates if downstream AGC is enabled. True means AGC is enabled. If the value is false, AGC is not enabled.

7.2.2.4.3. DsAgcPilotLossProtection

Provides an indication of the type of downstream AGC pilot loss protection in use.

other(1) – a vendor specific mechanism.
unknown(2) - the method of pilot loss protection is now known.
alternatePilotFreq(3) – using an alternate pilot frequency.
thermal(4) – using a type of thermal level control.
lockSettings(5) – locking the settings to the states in use when the pilot signal was lost.

7.2.2.4.4. DsAgcToUpperLimit

An indication of the amount of AGC RF level control available before the value hits the upper limit of RF level control.

7.2.2.4.5. DsAgcToLowerLimit

An indication of the amount of AGC RF level control available before the value hits the lower limit of RF level control.

7.2.2.5. BiDirLogicalPortStatus

This class points to the LogicalPortStatus class.

7.2.2.6. LogicalPortStatus

This class provides information on the status of the RF port.

Table 60 – LogicalPortStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
VendorDesc	AdminString	ro			
OperStatus	OperStatusType	ro	other(0), unknown(1), up(2), down(3)		

7.2.2.6.1. VendorDesc

This is a vendor specific description of the RF port status.

7.2.2.6.2. OperStatus

This attribute reports the current operational status of the RF port.

other(0) - the status is a supplier-specific state.
unknown(1) - the status is unknown.
up(2) - the status is operational.
down(3) - the status is not operational with no further information.

7.2.2.7. TestPointStatus

This class provides information on the output of an RF test point. The RF test point is described in [SCTE 279].

Table 61 – TestPointStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
PointIndex	UnsignedInt	ro			
PointFreq	UnsignedInt	ro		MHz	
PointLevel	TenthdBmV	ro		dBmV	

7.2.2.7.1. Point Index

The index of the test point, starting with test point 1 and continuing to RfTestPointNumPoints.

7.2.2.7.2. PointFreq

This attribute provides the frequency of the RF test point.

7.2.2.7.3. PointLevel

This attribute provides the RF level at the PointFreq of the RF test point. This level is the level at the test point, and [SCTE 279] states that the test point level is -20 dB relative to the associated input/output port.

7.2.3. RF Configuration (RfCfgGroup)

Figure 21 defines the RF configuration related classes and signals. These are classes have read-write attributes.

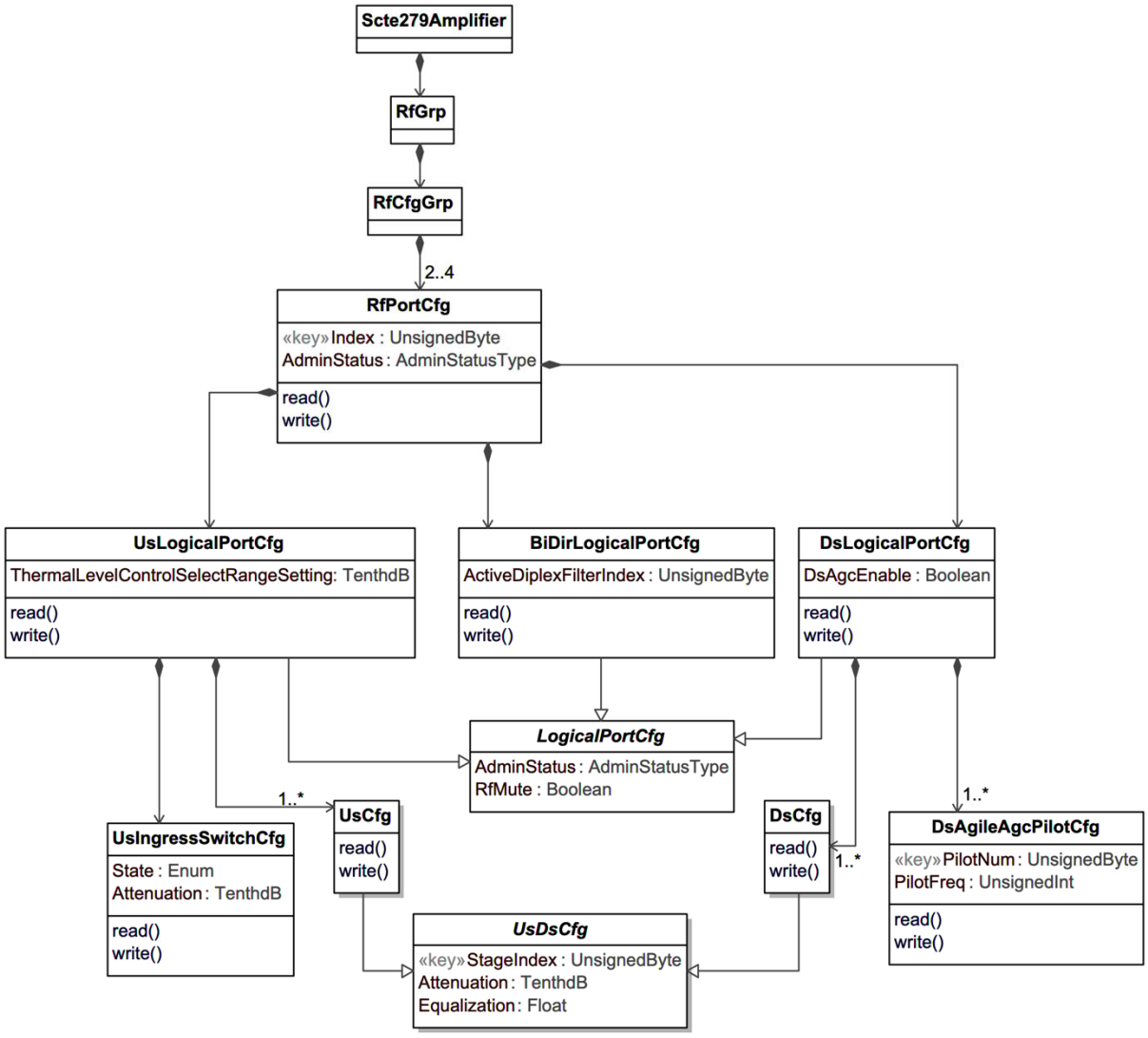

Figure 21 – RF Group Configuration Class Diagram

7.2.3.1. RfPortCfg

RfPortCfg allows the administrative status of the entire physical RF port to be set.

Table 62 – RfPortCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Index	UnsignedByte	rw			
AdminStatus	AdminStatusType	rw	up(1), down(2)		up(1)

7.2.3.1.1. Index

The index for the physical port. This Index is in the range of NumRfPorts from the RfCapabilities class.

7.2.3.1.2. AdminStatus

This attribute reports the state of the physical interface. When the Amplifier initializes, all interfaces start with AdminStatus in the up(1) state. As a result of either explicit management action or per configuration information retained by the managed system, AdminStatus either remains in the up(1) state or is changed to the down(2) state.

7.2.3.2. BiDirLogicalPortCfg

The BiDirLogicalPortCfg class is used to select the active diplex filter.

Table 63 – BiDirLogicalPortCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
ActiveDiplexFilterIndex	UnsignedByte	rw			

7.2.3.2.1. ActiveDiplexFilterIndex

This attribute indicates the active diplex filter, and is an index defined in DiplexFilterCapabilities in the RfCapabilities Group.

7.2.3.3. LogicalPortCfg

LogicalPortCfg allows administrative configuration of either an upstream logical port or a downstream logical port, including muting that port.

Table 64 – LogicalPortCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
AdminStatus	AdminStatusType	rw	up(1), down(2)		up(1)
RfMute	Boolean	rw			False

7.2.3.3.1. AdminStatus

This attribute reports the state of the logical port. When the Amplifier initializes, all interfaces start with AdminStatus in the up(1) state. As a result of either explicit management action or per configuration

information retained by the managed system, AdminStatus either remains in the up(1) state or is changed to the down(2) state.

7.2.3.3.2. RfMute

Muting a port means the output is at least 60 dB below the unmuted power of that port across the entire range of active frequencies.

Muting a logical port affects only the output power and does not impact the operational status of the port.

7.2.3.4. UsLogicalPortCfg

If the Amplifier supports upstream RF level control using Thermal level control with selectable range, the amount of level control is set here.

Table 65 – UsLogicalPortCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
ThermalLevelControlSelectRangeSetting	TenthdB	rw			

7.2.3.4.1. ThermalLevelControlSelectRangeSetting

The value of ThermalLevelControlSelectRangeSetting is governed by the UsLogicalPortCapabilities, specifically the values for:

ThermalLevelControlSelectRangeMinValue
ThermalLevelControlSelectRangeMaxValue
ThermalLevelControlSelectRangeStepSize

7.2.3.5. UsIngressSwitchCfg

This object controls the upstream ingress switch configuration. An upstream ingress switch is typically a three-state switch on each upstream input port that is user selectable to assist in ingress localization efforts.

Table 66 – UsIngressSwitchCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
State	Enum	rw	on(1), off(2), attenuated(3)		on(1)
Attenuation	TenthdB	rw			0 dB

7.2.3.5.1. State

This object controls the configuration of the upstream ingress switch.

The three states for the upstream RF path are:

on(1) = no extra attenuation added
off(2) = a very high amount of attenuation added
attenuated(3) = either a fixed or a user selected amount of additional attenuation is added

7.2.3.5.2. Attenuation

As indicated in RfCapabilities, UsLogicalPortCapabilities, if RfLevelControlType is thermalLevelControlSelect(5), then a selectable amount of attenuation is available. The amount of attenuation is also listed in UsLogicalPortCapabilities, specifically the values in:

ThermalLevelControlSelectRangeMinValue
ThermalLevelControlSelectRangeMaxValue
ThermalLevelControlSelectRangeStepSize

7.2.3.6. DsLogicalPortCfg

If the Amplifier supports downstream automatic gain control, the pilot frequencies are configured here.

Table 67 – DsLogicalPortCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
DsAgcEnable	Boolean	rw			False

7.2.3.6.1. DsAgcEnable

This attribute allows configuration of downstream automatic gain control. True means AGC is enabled. False means AGC is disabled.

If DsAgcEnable is True, the pilot frequencies are configured in DsAgileAgcPilotCfg.

At least one pilot frequency must be configured to set DsAgcEnable to True. The number of pilot frequencies available to be configured is defined in DsLogicalPortCapabilities:AgcNumAgilePilots.

7.2.3.7. DsAgileAgcPilotCfg

DsAgileAgcPilotCfg is used to configure downstream pilot frequencies.

Table 68 – DsAgileAgcPilotCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
PilotNum	UnsignedByte	rw			
PilotFreq	UnsignedInt	rw			

7.2.3.7.1. PilotNum

This key is the number of the pilot frequency. This key starts at 1 and the upper bound is defined in DsLogicalPortCapabilities:AgcNumAgilePilots.

7.2.3.7.2. PilotFreq

This is the frequency of the downstream pilot. This value must be within the range of MinDsFreq and MaxDsFreq. Attempting to set a pilot frequency outside the range of MinDsFreq and MaxDsFreq returns an error.

7.2.3.8. UsDsCfg

UsDsCfg allows configuration of the attenuation and equalization stages.

Table 69 – UsDsCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
StageIndex	UnsignedByte	rw			
Attenuation	TenthdB	rw			
Equalization	Byte	rw			

7.2.3.8.1. StageIndex

This attribute indicates the number of the stage index. This value starts at 1 and is bounded by either NumUsStages or NumDsStages.

7.2.3.8.2. Attenuation

If this is an attenuation stage as indicated in UsDsStageCapabilities:Type, this attribute configures the Attenuation associated with this stage.

If this is an equalization stage as indicated in UsDsStageCapabilities:Type, then attempting to set this attribute returns an error.

The allowable ranges of Attenuation settings are described in UsDsStageCapabilities in:

MinAttenuation
MaxAttenuation
AttenuationStepSize

7.2.3.8.3. Equalization

If this is an equalization stage as indicated in UsDsStageCapabilities:Type, this attribute configures the Attenuation associated with this stage. This value can be a negative number.

If this is an attenuation stage as indicated in UsDsStageCapabilities:Type, then attempting to set this attribute returns an error.

The allowable ranges of equalization settings are described in UsDsStageCapabilities in:

MinEqualization
MaxEqualization
EqualizationStepSize

7.3. Networking Group (NetworkingGrp)

Figure 22 defines the Networking configuration related classes and signals. These are classes have read-only attributes.

The Amplifier ***shall*** implement all classes described in the NetworkingGrp information model.

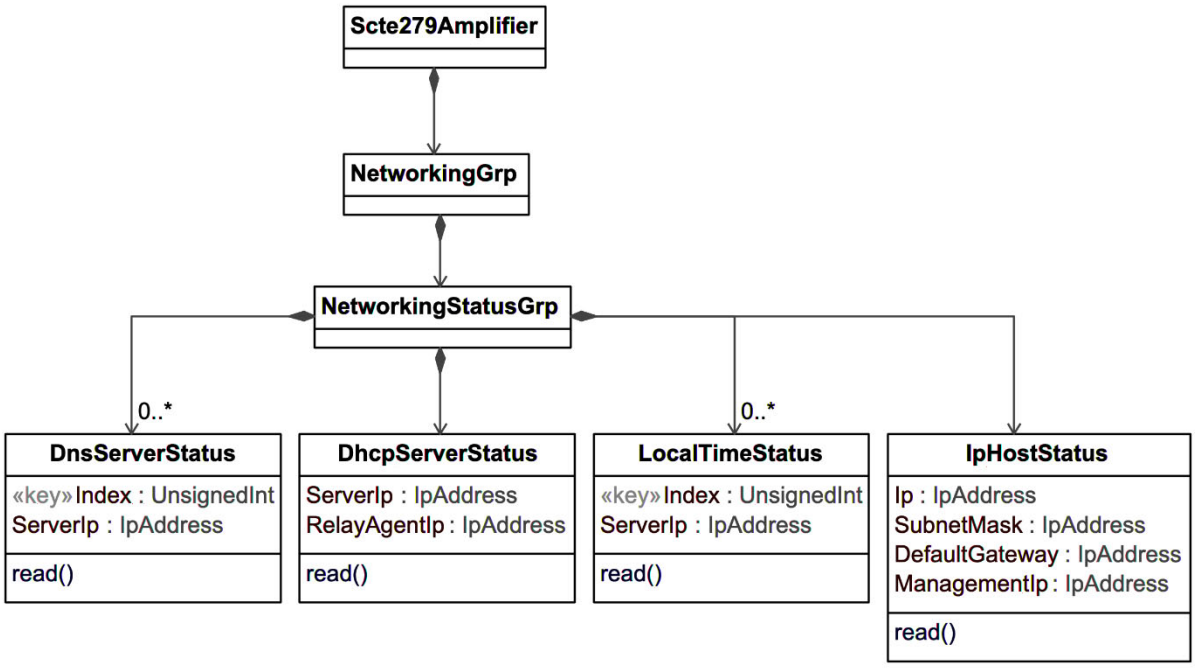

Figure 22 – Networking Group Component Diagram

7.3.1. IpHostStatus

The IpHostStatus object identifies IP addresses and information associated with the Amplifier.

Table 70 – IpHostStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Ip	IpAddress	ro			
SubnetMask	IpAddress	ro			
DefaultGateway	IpAddress	ro			
ManagementIp	IpAddress				

7.3.1.1. Ip

This attribute reports the IP address used by the eAMP.

7.3.1.2. SubnetMask

This attribute reports the subnet mask or prefix length associated with Ip.

7.3.1.3. DefaultGateway

This attribute reports the IP address of the **default gateway** on the network that serves as the forwarding host (router) to other networks when no other route specification matches the destination IP address of a packet.

7.3.1.4. ManagementIp

This attribute reports the IP address of the Amplifier manager.

7.3.2. LocalTimeStatus

The LocalTimeStatus object identifies time servers used by the Amplifier.

Table 71 – LocalTimeStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Index	UnsignedInt	ro			
ServerIp	IpAddress	ro			

7.3.2.1. Index

This attribute reports an index for a time server used by the eAMP.

7.3.2.2. ServerIp

This attribute reports the IP address of the server referenced by Index.

7.3.3. DhcpServerStatus

The DhcpServerStatus object identifies DHCP servers used by the Amplifier.

Table 72 – DhcpServerStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
ServerIp	IpAddress	ro			
RelayAgentIp	IpAddress	ro			

7.3.3.1. ServerIp

This attribute reports the IP address of the DHCP server used by the eAMP.

7.3.3.2. RelayAgentIp

This attribute reports the IP address of the DHCP relay server used by the eAMP.

7.3.4. DnsServerStatus

The DnsServerStatus object identifies DNS servers used by the Amplifier.

Table 73 – DnsServerStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
Index	UnsignedInt	ro			
ServerIp	IpAddress	ro			

7.3.4.1. Index

This attribute reports an index for a DNS server used by the eAMP.

7.3.4.2. ServerIp

This attribute reports the IP address of the server referenced by Index.

7.4. PNM Group (PnmGrp)

The Proactive Network Maintenance (PNM) Group describes a method for the Amplifier to perform a spectrum capture file.

The Amplifier *may* implement all classes described in the PnmGrp information model.

7.4.1. Spectrum Capture

The Amplifier *may* implement the Spectrum Capture feature.

This group of objects provides a spectrum capture function on a physical port and can be either upstream or the downstream. Depending on the physical port and the direction, the measurement is generally either before or after the attenuation and equalization stages. The spectrum capture feature can be useful for checking signal levels as well as setting up amplifiers for the correct levels.

The spectrum capture amplitude data ***shall*** be reported as the signal level at the physical port on the amplifier.

This implementation is borrowed from both [CM-OSSIv4.0] and [PNM-3.1]. Please see these references for additional information on the spectrum capture feature. Even though only the spectrum capture test is included in this standard, the DOCSIS PNM structure is being included in the event in the future additional DOCSIS PNM tests are included with the Amplifier.

Each measurement is a data collection event that provides the energy content of the signal at each frequency within a specified range. The result of a measurement is a table consisting of one or more rows. Each row corresponds to a capture of spectral data across a specified segment bandwidth. The frequency range of each segment is divided into bins, which are a discrete set of evenly spaced frequencies across the band. The width of each bin (resolution bandwidth) is generally equal to or slightly greater than the spacing between bins. Each bin has an associated amplitude value in the table, which represents the amount of energy measured in that frequency bin. The segments are constrained to be contiguous; that is, the start frequency of each segment equals the end frequency of the previous segment plus the bin spacing. Thus, the concatenation of all segments results in a wideband spectral capture. The measurement table is updated at a rate that is vendor specific.

7.4.2. PNM Common Class Diagram

Figure 23 defines the Proactive Network Maintenance (PNM) Common related classes and signals (notifications) and includes PnmStatus and PnmResult.

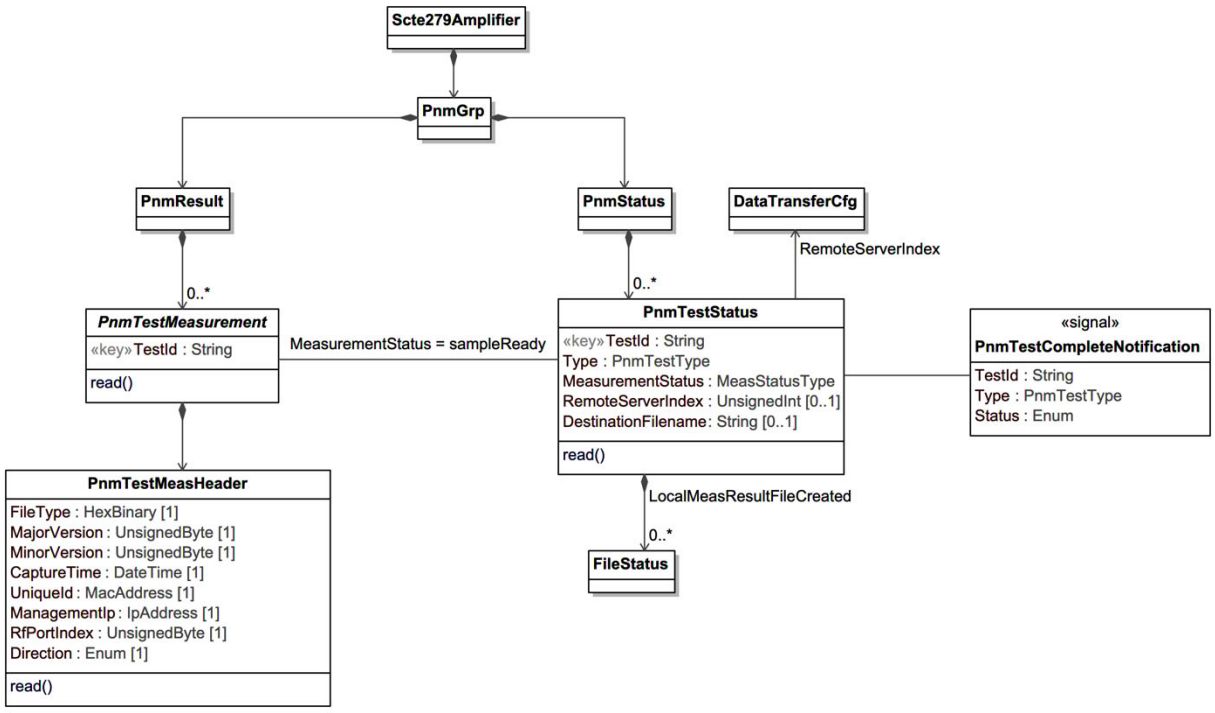

Figure 23 – PNM Common Class Diagram

7.4.2.1. PnmStatus

The PnmStatus class is the container for all status related to PNM test common functionality.

7.4.2.1.1. DataTransferCfg

The DataTransferCfg class is defined in Section 7.1.9.4.

7.4.2.1.2. FileStatus

The FileStatus class is defined in Section 7.1.9.2.

7.4.2.1.3. PnmTestStatus

The PnmTestStatus class reports the status of historical, active, and scheduled PNM test instances.

Table 74 – PnmTestStatus Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
TestId	String	ro			
Type	Enum	ro	other(0), usSpectrumCapture(1), dsSpectrumCapture(2)		
MeasurementStatus	Enum	ro	other(1), inactive(2), busy(3), sampleReady(4), error(5), resourceUnavailable(6), sampleTruncated(7), interfaceModification(8)		
RemoteServerIndex	UnsignedInt	ro			
DestinationFilename	String				

7.4.2.1.3.1. TestId

This key attribute is a string uniquely identifying a PNM test instance.

7.4.2.1.3.2. Type

This attribute reports the type of spectrum capture test the MeasurementStatus applies to.

7.4.2.1.3.3. MeasurementStatus

This attribute reports the status of the PNM test instance. The MeasStatusType values are interpreted as follows:

other(1) - Indicates any state not described below
inactive(2) - Indicates that a test is not started or in progress
busy(3) - Indicates that a test has been started and is in progress
sampleReady(4) - Indicates that a test has completed and that the measurement data is ready
error(5) - Indicates that there was an error starting or during the test and any test data, if available, may not be valid
resourceUnavailable(6) - Indicates that the test could not be started due to lack of test platform resources
sampleTruncated(7) - Indicates that the size of the requested data exceeded file size supported
interfaceModification(8) - Indicates that the interface numbering is changed due to DBC message or when Primary backup is changed

7.4.2.1.3.4. RemoteServerIndex

This attribute uniquely identifies a destination for PNM test result measurements. This attribute refers to an instance of the DataTransferCfg object. Note that the RemoteServerIndex attribute of the DataTransferCfg object is required to exist before provisioning the corresponding value in this attribute.

If this attribute is not populated or set to zero, the Amplifier will create a local file or files for the results.

If the attribute is set to a non-zero value, the Amplifier uses the instance of DataTransferCfg defined by the RemoteServerIndex to determine how to handle the results file or files.

7.4.2.1.3.5. DestinationFilename

This attribute identifies the destination filename for PNM test result measurements. This attribute is an extension to an instance of the DataTransferCfg object, defining a filename.

7.4.2.1.4. PnmTestCompleteNotification

PnmTestCompleteNotification is an asynchronous notification informing the PNM Server about the status of the completed, failed, or aborted PNM test instance. This notification is sent by the Amplifier when a PNM test terminates.

Table 75 – PnmTestCompleteNotification Object Attributes

Attribute Name	Type	Required	Type Constraints	Units	Default Value
TestId	String	yes			
Type	Enum	yes	other(0), usSpectrumCapture(1), dsSpectrumCapture(2)		
Status	Enum	yes	other(0), success(1), aborted(2), error(3)		

7.4.2.1.4.1. TestId

This key attribute is a string uniquely identifying a PNM test instance. See the object definition of TestStatus for a definition of the TestId format.

7.4.2.1.4.2. Type

This attribute reports the type of PNM test the status applies to.

usSpectrumCapture is an upstream spectrum capture.
dsSpectrumCapture is a downstream spectrum capture.

7.4.2.1.4.3. Status

This attribute reports the state of the completed PNM test instance.

other(0) is provided for vendor-specific status.
success(1) indicates the PNM test instance completed successfully and results are available and/or have been uploaded.
aborted(2) indicates the PNM test instance was aborted and test results were discarded.
error(3) indicates an error occurred during the execution of the PNM test instance and the test results were discarded.

7.4.2.2. PnmResult

The PnmResult class is the top-level container for all test measurements related to PNM test functionality.

7.4.2.2.1. PnmTestMeasurement

The PnmTestMeasurement class reports the measurement results of PNM test instances. This is an abstract class where each of the different PNM tests realize this class through inheritance/specialization. This class is instantiated each time a PNM test is executed and the MeasurementStatus attribute contains a value of 'sampleReady' for the PnmTestStatus object.

Table 76 – PnmTestMeasurement Object Attributes

Attribute Name	Type	Required	Type Constraints	Units	Default Value
TestId	String	yes			

Table 77 – PnmTestMeasurement Object Associations

Attribute Name	Type	Near-end Multiplicity	Far-end Multiplicity	Label
PnmTestMeasHeader	Directed composition to PnmTestMeasHeader	1	1	

7.4.2.2.1.1. TestId

This key attribute is a string uniquely identifying a PNM test instance.

7.4.2.2.2. PnmTestMeasHeader

This object contains the common PNM header information for reporting measurements results. See Section 7.4.3.2.2.2 for details on formatting the file.

Table 78 – PnmTestMeasHeader Object Attributes

Attribute Name	Type	Required	Type Constraints	Units	Default Value
FileType	HexBinary	No	SIZE(4)		
MajorVersion	UnsignedByte	No			
MinorVersion	UnsignedByte	No			
CaptureTime	UnsignedInt	Yes			
UniqueId	MacAddress	Yes			
ManagementIp	IpAddress	No			
RfPortIndex	UnsignedByte	Yes			
Direction	Enum	Yes	upstream(1), downstream(2)		

7.4.2.2.2.1. FileType

A four-byte hexadecimal identifier specific to the type of PNM test that generated the data file. Refer to the [CANN] DOCSIS PNM Registry for defined File Types.

Spectrum Capture uses one of the following FileType values.

504E4D09 – when the file does not contain the MajorVersion and MinorVersion header elements.

504E4E09 – when the file does contain the MajorVersion and MinorVersion header elements.

7.4.2.2.2.2. MajorVersion

This attribute represents the file header version. This value is incremented by one when the header format is modified by this specification.

7.4.2.2.2.3. MinorVersion

This attribute is reserved for vendor-specific and vendor-defined version information.

7.4.2.2.2.4. CaptureTime

This attribute represents the epoch time (also known as 'unix time') which is the number of seconds that have elapsed since midnight Coordinated Universal Time (UTC), Thursday, 1 January 1970.

When the Amplifier creates a capture file with multiple result sets, the value of CaptureTime corresponds to the first result set in the capture file.

7.4.2.2.2.5. UniqueID

UniqueId is a lower-case hexadecimal value (i.e., "12:34:56:78:ab:cd"), representing the MAC address of the Amplifier. This is the same MAC Address used in the SystemStatus class.

7.4.2.2.2.6. ManagementIp

This is the IP Address of the Amplifier Manager. This is the same IP Address used in the IpHostStatus class.

7.4.2.2.2.7. RfPortIndex

This is the physical port on which the spectrum capture occurred.

7.4.2.2.2.8. Direction

This represents if the spectrum capture is for the upstream or the downstream spectrum.

7.4.3. PNM Spectrum Capture Component Diagram

The spectrum capture feature described here is associated with the eAMP logical entity. The transponder could also have a spectrum capture capability, however, that is not described here.

Figure 24 illustrates the PNM Spectrum Capture component diagram including the eAMP (Server) and PNM Server (Client) components for the common PNM test operations. The eAMP Server component provides an operations/methods interface that contains operations that are invoked by the PNM Server Client to perform the actions.

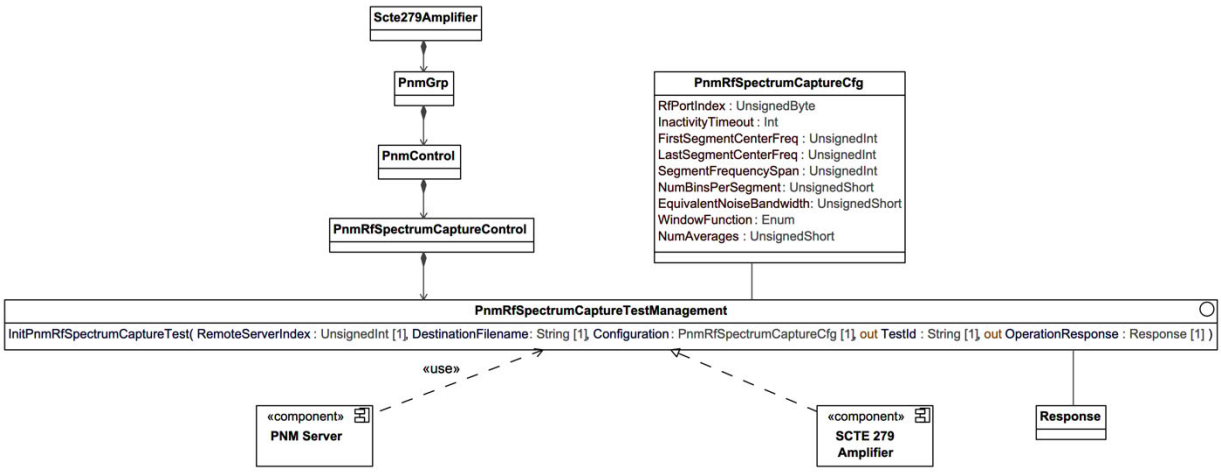

Figure 24 – Spectrum Capture Component Diagram

Note that is possible for the spectrum capture command to be invoked multiple times in short order to request spectrum captures on multiple ports. It is up to the implementation to handle multiple concurrent requests in an orderly fashion, for example, without overwriting results files.

7.4.3.1. PnmRfSpectrumCaptureCfg

This object is used to configure the frequency spectral capture test in the Amplifier.

Table 79 – PnmRfSpectrumCaptureCfg Object Attributes

Attribute Name	Type	Access	Type Constraints	Units	Default Value
RfPortIndex	UnsignedByte	rw			
InactivityTimeout	Int	rw	0..86400	seconds	300
FirstSegmentCenterFreq	UnsignedInt	rw		Hz	
LastSegmentCenterFreq	Unsigned	rw		Hz	
SegmentFrequencySpan	UnsignedInt	rw		Hz	0
NumBinsPerSegment	UnsignedShort	rw	2..2048	Bins-per-segment	256
EquivalentNoiseBandwidth	UnsignedShort	rw	50..500	Hundredths of bin spacing	150
WindowFunction	Enum	rw	other(0), hann(1), blackmanHarris(2), rectangular(3), hamming(4), flatTop(5), gaussian(6), chebyshev(7)		
NumAverages	UnsignedShort	rw	1..1000		1

7.4.3.1.1. RfPortIndex

This is the physical port on which the spectrum capture is to occur.

7.4.3.1.2. InactivityTimeout

This attribute controls the length of time after the last spectrum capture measurement before the feature is automatically disabled. If set to a value of 0, the feature will remain enabled until it is explicitly disabled.

7.4.3.1.3. FirstSegmentCenterFreq

This attribute controls the center frequency of the first segment for the spectrum capture measurement.

The frequency bins for this segment lie symmetrically to the left and right of this center frequency. If the number of bins in a segment is odd, the segment center frequency lies directly on the center bin. If the number of bins in a segment is even, the segment center frequency lies halfway between two bins.

Note that the combination of FirstSegmentCenterFreq and LastSegmentCenterFreq can span the diplex filter. If this attribute is set to an invalid value, the Amplifier may return an error of inconsistentValue, or may adjust the value of the attribute to the closest valid value.

7.4.3.1.4. LastSegmentCenterFreq

This attribute controls the center frequency of the last segment of the spectrum capture measurement.

The frequency bins for this segment lie symmetrically to the left and right of this center frequency. If the number of bins in a segment is odd, the segment center frequency lies directly on the center bin. If the number of bins in a segment is even, the segment center frequency lies halfway between two bins.

Note that the combination of FirstSegmentCenterFreq and LastSegmentCenterFreq can span the diplex filter. If this attribute is set to an invalid value, the Amplifier may return an error of inconsistentValue, or may adjust the value of the attribute to the closest valid value.

7.4.3.1.5. SegmentFrequencySpan

This attribute controls the frequency span of each segment (instance) of the RF spectrum capture. If set to a value of 0, then a default span will be chosen based on the capabilities of the Amplifier. Segments are contiguous from the FirstSegmentCenterFrequency to the LastSegmentCenterFrequency and the center frequency for each successive segment is incremented by the SegmentFrequencySpan.

The number of segments is:

(LastSegmentCenterFrequency - FirstSegmentCenterFrequency)/SegmentFrequencySpan + 1.

The value of SegmentFrequencySpan affects the number of instances in the results file. A more granular SegmentFrequencySpan may increase the acquisition time of the test results and the size of the results file.

Note that if this attribute is set to an invalid value, the Amplifier may return an error of inconsistentValue, or may adjust the value of the attribute to the closest valid value.

7.4.3.1.6. NumBinsPerSegment

This attribute controls the number of bins collected by the measurement performed for each segment (instance) of the results file.

Note that if this attribute is set to an invalid value, the device may return an error of inconsistentValue, or may adjust the value of the attribute to the closest valid value.

7.4.3.1.7. EquivalentNoiseBandwidth

This attribute allows the user to request an equivalent noise bandwidth for the resolution bandwidth filter used in the spectrum capture. This corresponds to the spectral width of the window function used when performing a discrete Fourier transform for the capture.

The window function which corresponds to a value written to this attribute may be obtained by reading the value of the WindowFunction attribute.

If an unsupported value is requested, the device may return an error of inconsistentValue, or choose the closest valid value to the one which is requested. If the closest value is chosen, then a subsequent read of this attribute will return the actual value that is in use.

7.4.3.1.8. WindowFunction

This attribute controls or indicates the windowing function that will be used when performing the discrete Fourier transform for the capture. The WindowFunction and the EquivalentNoiseBandwidth are related. If a particular WindowFunction is selected, then the EquivalentNoiseBandwidth for the function in use will be reported by the EquivalentNoiseBandwidth attribute. Alternatively, if an EquivalentNoiseBandwidth value is chosen and a WindowFunction function representing that EquivalentNoiseBandwidth is defined in the Amplifier, that value will be reported in the WindowFunction object, or a value of 'other' will be reported. Use of "modern" windowing functions not yet defined will likely be reported as 'other'.

Note that all window functions may not be supported by all devices. If an attempt is made to set the attribute to an unsupported window function, or if writing of the WindowFunction object is not supported by an implementation, an error will be returned.

7.4.3.1.9. NumAverages

This attribute controls the number of averages that will be performed on spectral bins. The average will be computed using the "leaky integrator" method, where:

reported bin value = alpha*accumulated bin values + (1-alpha)*current bin value

Alpha is one minus the reciprocal of the number of averages. For example, if N=25, then alpha = 0.96. A value of 1 indicates no averaging. Re-writing the number of averages will restart the averaging process. If there are no accumulated values, the accumulators are made equal to the first measured bin amplitudes.

The number of averages will be set by writing NumAverages attribute. If an attempt is made to set the attribute to an unsupported number of averages, an error of inconsistentValue will be returned.

7.4.3.2. PnmRfSpectrumCaptureTestManagement

7.4.3.2.1. InitPnmRfSpectrumCaptureTest Operation

InitPnmRfSpectrumCaptureTest is a non-blocking asynchronous operation that initiates the spectrum capture test. A successful operation results in the Amplifier starting a spectrum capture collection. The Amplifier will return a TestId which uniquely identifies the specific test that was started.

When the Amplifier completes the test, the Amplifier will send the TestCompleteNotification signaling the test results have been uploaded to the RemoteServerIndex.

The Amplifier shall reject configuring a value for the RemoteServerIndex of the InitPnmRfSpectrumCaptureTest object if that value does not exist in the corresponding RemoteServerIndex attribute of the DataTransferCfg instance.

A successful operation will result in the Amplifier initiating a file upload using the configured file transfer method as described in Section 7.1.10.3.

Table 80 – PnmRfSpectrumCaptureTestManagement Operation Parameters

Parameter Name	Type	Type Constraints	Direction	Multiplicity	Units
RemoteServerIndex	UnsignedInt		In	1	
DestinationFilename	String		In	0..1	
Configuration			In	1	
TestId	String		Out	1	
OperationResponse	Response		Out	1	

7.4.3.2.2. PnmRfSpectrumCaptureTestManagement Parameter Definitions

7.4.3.2.2.1. RemoteServerindex

This parameter specifies the remote server defined in the DataTransferCfg class.

7.4.3.2.2.2. DestinationFilename

This optional parameter is a string with the name for the spectrum capture data file provisioned by the operator.

The DestinationFilename can only be changed while a test is not in progress. An attempt to set this value while the value of MeasStatusType is 'busy' will return 'inconsistentValue'.

If the value of this object is the DEFVAL (empty string), then a default filename value will be used. Otherwise, the value set will be used as the filename. If a default filename is generated, then that value will be returned in this object and will represent the filename that was used for the test. All subsequent tests should set this object to a meaningful value or to an 'empty string' value (to generate a new default filename) before starting a new test.

If a default filename value is used, it is generated as the test name plus the eAMP MAC Address plus the 'epoch time'. The epoch time (also known as 'unix time') is defined as the number of seconds that have elapsed since midnight Coordinated Universal Time (UTC), Thursday, 1 January 1970.

Hence, the format would be:

PNMSpecAnData_<eAMP MAC address>_<RfPortIndex>_<epoch>

For example: PNMSpecAnData_0010181A2D11_2_1403405123

The data file is composed of a header plus the Spectrum Capture Data. The header is composed of ordered fixed-length fields. Unless otherwise specified, the header fields contain hex values that are right justified within the field. If necessary, the field is left-padded with zero values.

7.4.3.2.2.3. Configuration

This parameter is the complex set of configurations for the Spectrum Capture PNM test operation.

7.4.3.2.2.4. TestId

This parameter is a string uniquely identifying the spectrum capture PNM test instance.

7.4.3.2.2.5. OperationResponse

This parameter is the response to the PnmRfSpectrumCaptureTestManagement command.

Table 81 – PnmRfSpectrumCaptureTestManagement Response Object Attributes

Attribute Name	Type	Required Attribute	Type Constraints	Units
Success	Boolean	Yes		
ErrorTag	Enum	No		
ErrorMessage	String	No		

Success
This attribute reports the success or failure of the operation. True indicates the action performed by the operation was successful.

ErrorTag
If the operation was unsuccessful, this attribute reports the Error Tag for the operation. This attribute is not included if the action was successful.

ErrorMessage
If the operation was unsuccessful, this attribute reports the Error Message for the operation. This attribute is not included if the action was successful.

Table 82 – PnmRfSpectrumCaptureTestManagement Response Operation Errors

ErrorTag	ErrorMessage
Entity Not Found	Filename does not exist on this device
Not in Valid State	Reset is in progress
Internal Error	The device had an error and could not process the request
Invalid Input	Invalid input parameter
Access denied	The operation request is not authorized
Operation Not Supported	The device does not support the operation or feature

7.4.4. PNM Common Component Diagram

The PNM Common component diagram illustrates the eAMP (Server) and PNM Server (Client) components for the common PNM test operations. The eAMP Server component provides an operations/methods interface that contains operations that are invoked by the PNM Server Client to perform the actions.

7.4.4.1. PnmTestManagement Component Diagram

Figure 25 illustrates the PnmTestManagement component diagram including the eAMP (Server) and PNM Server (Client) components for the common PNM operations.

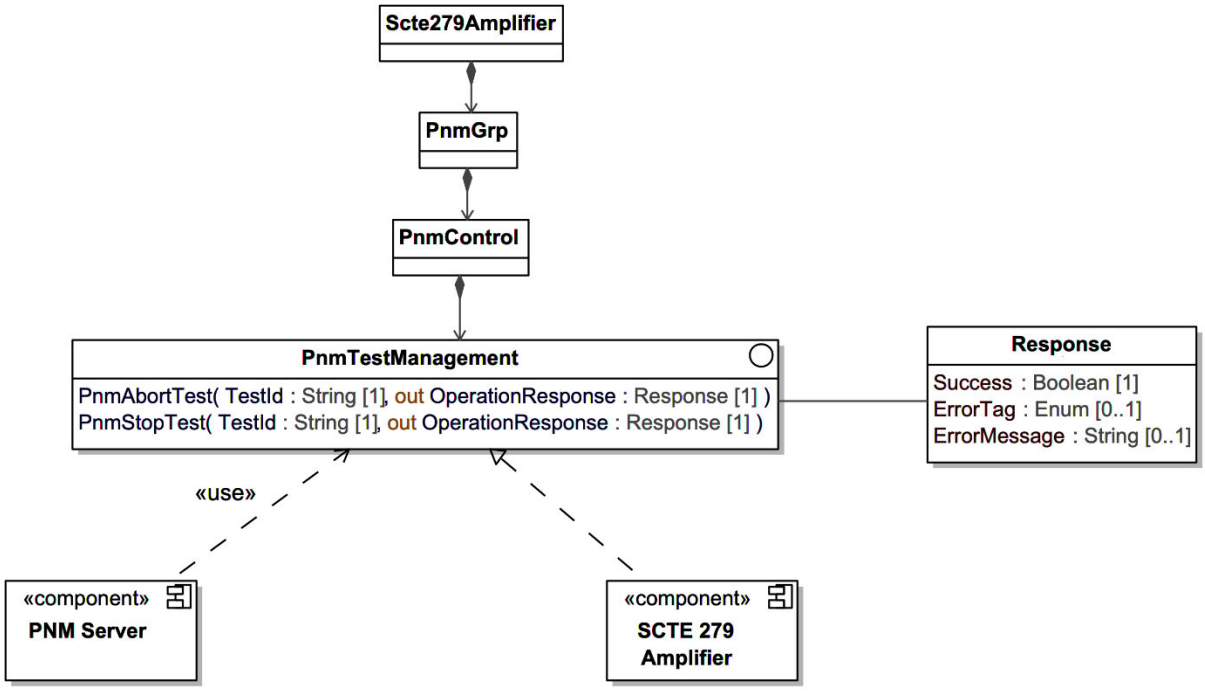

Figure 25 – PnmTestManagement Component Diagram

7.4.4.1.1. PnmAbortTestOperation

PnmAbortTest is a non-blocking asynchronous operation that terminates execution of specific running PNM test instances. The Amplifier is not required to provide measurement results after being invoked with the PnmAbortTest operation.

A successful operation will result in the Amplifier ceasing sampling for the PNM test instance specified by the TestId, discarding captured measurement data, sending a response with value ‘success’, and terminating all actions of the PNM test instance.

Table 83 – PnmAbortTest Operation Parameters

Parameter Name	Type	Type Constraints	Direction	Multiplicity	Units
TestId	String		In	1	
OperationResponse	Response		Out	1	

7.4.4.1.1.1.1. PnmAbortTest Parameter Definitions

TestId

This attribute is a string uniquely identifying a PNM test instance.

OperationResponse

This parameter is the response to the PnmAbortTest command.

Table 84 – PnmAbortTest Response Object Attributes

Attribute Name	Type	Required Attribute	Type Constraints	Units
Success	Boolean	Yes		
ErrorTag	Enum	No		
ErrorMessage	String	No		

Success
This attribute reports the success or failure of the operation. True indicates the action performed by the operation was successful.

ErrorTag
If the operation was unsuccessful, this attribute reports the Error Tag for the operation. This attribute is not included if the action was successful.

ErrorMessage
If the operation was unsuccessful, this attribute reports the Error Message for the operation. This attribute is not included if the action was successful.

Table 85 – PnmAbortTest Operation Errors

ErrorTag	ErrorMessage
Entity Not Found	Filename does not exist on this device
Not in Valid State	Reset is in progress
Internal Error	The device had an error and could not process the request
Invalid Input	Invalid input parameter
Access denied	The operation request is not authorized
Operation Not Supported	The device does not support the operation or feature

7.4.4.1.2. PnmStopTestOperation

PnmStopTest is a non-blocking asynchronous operation that terminates execution of specific running PNM test instances. The PnmStopTest operation is a graceful exit of a PNM Test where the Amplifier provides the measurement results from the test.

A successful operation will result in the Amplifier ceasing sampling for the PNM test instance specified by the TestId, saving captured measurement data, sending a response with value 'success', and terminating all actions of the PNM test instance.

Table 86 – PnmStopTest Operation Parameters

Parameter Name	Type	Type Constraints	Direction	Multiplicity	Units
TestId	String		In	1	
OperationResponse	Response		Out	1	

7.4.4.1.2.1.1. PnmStopTest Parameter Definitions

TestId

This attribute is a string uniquely identifying a PNM test instance.

OperationResponse

This parameter is the response to the PnmStopTest command.

Table 87 – PnmStopTest Response Object Attributes

Attribute Name	Type	Required Attribute	Type Constraints	Units
Success	Boolean	Yes		
ErrorTag	Enum	No		
ErrorMessage	String	No		

Success
This attribute reports the success or failure of the operation. True indicates the action performed by the operation was successful.

ErrorTag
If the operation was unsuccessful, this attribute reports the Error Tag for the operation. This attribute is not included if the action was successful.

ErrorMessage
If the operation was unsuccessful, this attribute reports the Error Message for the operation. This attribute is not included if the action was successful.

Table 88 – PnmStopTest Operation Errors

ErrorTag	ErrorMessage
Entity Not Found	Filename does not exist on this device
Not in Valid State	Reset is in progress
Internal Error	The device had an error and could not process the request
Invalid Input	Invalid input parameter
Access denied	The operation request is not authorized
Operation Not Supported	The device does not support the operation or feature

8. Security Considerations

This section provides an informative discussion on how security applies to smart amplifiers.

8.1. The Need For Security

8.1.1. Attack Vectors

Figure 26 shows possible methods of attack (vectors) for the Amplifier. Adversaries can choose to either remotely attack exposed interfaces at managed HFC network elements or by directly connecting to the Amplifier. Adversaries may attempt to access the management network from the consumer side of the HFC network or from the operator and Internet side. This may be done from compromised peers within the HFC network or management network; or it may be done from external sources such as the customer network. Direct access to the Amplifier can leverage an exposed hardware interface or wireless interface.

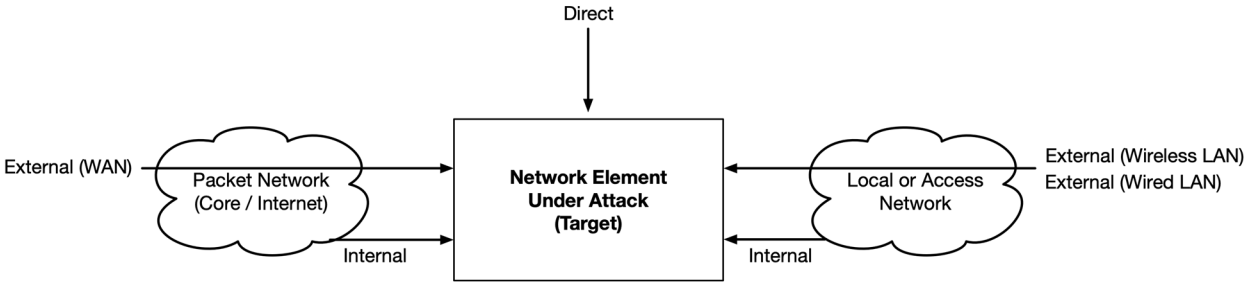

Figure 26 – Attack Vectors

If an adversary can access a vulnerable amplifier, they may be able to disrupt service, change device settings, access other devices, or achieve other nefarious outcomes.

One attack vector that requires particular attention here is software supply chain attacks. Firmware updates for an amplifier are provided through a complex supply chain and insecure software images can be compromised at many points in that supply chain. Tampered or unauthorized firmware can change the behavior of an amp; or it can simply provide for unauthorized access.

8.1.2. Attack Motivations

There are many motivations that may drive adversaries to attack amplifiers. These may include:

- Nation state attacks to disrupt critical infrastructure,
- Ransom attacks against the operator that threaten disruption or destruction of HFC network infrastructure,
- Denial of Service attacks such as interrupting Internet of things functionality in a target community to disrupt emergency service calls or prevent alerts and alarms from home and business surveillance systems,
- Damage the brand of the operator.

Once an Amplifier is compromised, it may be used as a pivot to attack other assets in the network such as Distributed Access Architecture (DAA) elements, fiber nodes, and back-office systems.

Motivations for such attacks include those listed above and the following:

- Theft of service (by changing settings on other devices such DAA components),
- Denial of service attacks against specific customers (gamer-paid attacks are common),
- Privilege escalation (by changing settings on other devices such DAA components),
- Access video distribution infrastructure for content piracy,
- Leverage servers as compute / storage resources (crypto mining, illicit content, etc.).

8.2. Mitigating Controls

Traditionality, network security relies on a trust boundary based on private addresses, routing controls, and filtering to prevent unauthorized access to managed devices. These traditional controls may be insufficient in today's complex architectures that are continuously under threat from bad actors. Simply relying on authentication of the amplifier's transponder and provisioning it on a private IP subnet (e.g., 10.x.x.x) will not prevent all attack vectors. Rather, a Zero Trust Architecture (ZTA) approach is recommended. ZTA assumes no trust domain boundary. References for ZTA include [ZTA-NIST] and [ZTA-EU].

The principals of ZTA can be applied to an HFC network by establishing security associations between functions and devices rather than simply interfaces. Such a security association can be:

- Strong identity: Identity is the basis for any meaningful trust system. Identity should be based on a secret paired with a unique identifier. The identity is attested by a certificate or equivalent by signing or equivalent cryptographic operation. The certificate can contain other information (but not any information that should be changed such as software versions).
- Authenticated: Each security association is verified when the association is requested using a cryptographic challenge.
- Authorized: Once entities have validated their mutual identities, their resource or activity accesses are still be authorized. Authorization can be based on a system or service wide policy system. The policy system can assume a least privilege orientation and assure separation of duty and function. Implementation may use a policy lookup or token grant approach.
- Isolated: Isolation of network, storage, and compute resources used for specific workloads is a goal. There are a wide range of obvious security risks that are managed this way; however, it is equally important from a performance perspective. Specifically, workloads or process should not impact other workloads or processes unless allowed by the operator. Isolation may be achieved by network segmentation (through secure addressing or encapsulation) and various virtualization tools for ensuring workload isolation in memory, CPU, and storage.
- Confidentiality: Data and communications should be kept private. The isolation functions discussed above may achieve sufficient confidentiality; however, encryption will ensure even stronger confidentiality, assuming adequate protection of encryption keys.
- Attested: All the security controls that implement a security association and protect it must be provably untampered. This is traditionally done using accounting and logging mechanisms. There are improvements in trusted computing systems that allow secure boot and run time monitoring to improve on legacy approaches. Whatever specific strategies are used, the goal must be to verify that the infrastructure and the security associations implemented to interconnect both hardware and software components are, indeed, what they are expected to be.

8.3. Smart Amplifier Security Model

From a management perspective, an Amplifier consists of four functional components.

- the physical amplifier,
- management of that amplifier,
- a transponder,

- local management which must include a wired interface (such as a USB port) and may include wireless access (such as Bluetooth or Wi-Fi).

Figure 27 presents a security view of the Amplifier including the transponder.

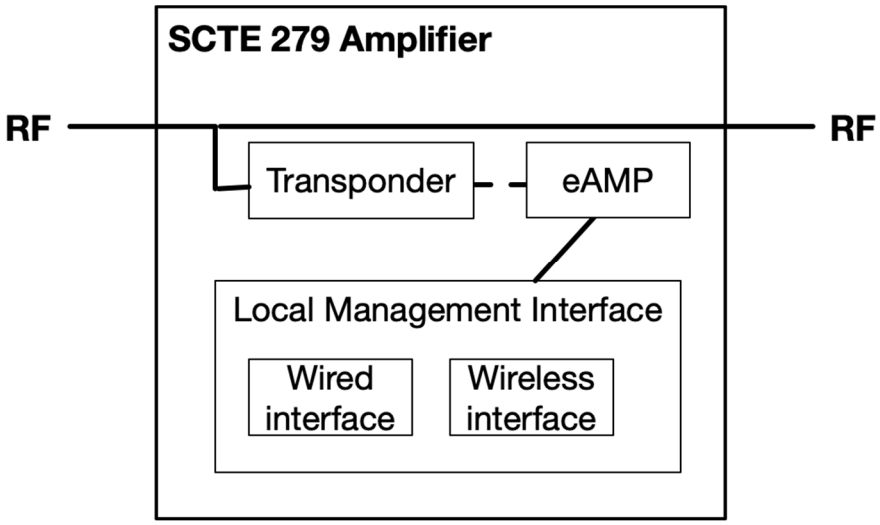

Figure 27 – Amplifier Security Reference Model

It is important to note that on an HFC network, the RF signal is bidirectional, and the transponder is "in-band". The transponder may not authenticate to the network (or equivalent DAA device). Authentication is for the link layer and applies only to the ability to attach to the network – it is not an authorization to communicate on that network which relies on higher level functions in the network. And finally, the firmware for the transponder and eAMP may, or may not, be distinct.

Access to Amplifier functions can only be allowed via interfaces specifically prescribed by [SCTE 279] and other SCTE standards.

The following subsections apply ZTA principles to the Amplifier.

8.3.1. Local Management Interface

The local management interface is associated with the eAMP.

Management access interfaces, also called console ports, are communications paths (usually RS-232, but could be Ethernet, etc.) and debug software that interact with a user. The local management interface can have built-in security so that connection can only be established by authorized persons. Access using this interface can be authenticated and authorized using multi-factor authentication. Access attempts and activities can be logged and reported to the appropriate management servers. The functions exposed to this interface can only enable configuration of the amplifier.

It is common practice that the local craft interface can be disabled remotely. However, in the event the managed amplifier loses connectivity to the management network, an amplifier may be "bricked" if the local craft interface is disabled. It is important that a reset of the amplifier be possible to allow local management access to the amplifier.

Local management can optionally be conducted with an additional wireless connection. If a wireless option is provided, the wireless transceiver would be plugged into the local management interface. As with a physical port, wireless interfaces can be authenticated and authorized, and use can be logged and reported.

8.3.2. Remote Amplifier Management

The information model assumes using RESTCONF to access a YANG data model. RESTCONF uses HTTPS, which in turn leverages Transport Layer Security (TLS), specifically either TLS version 1.2 or TLS version 1.3. In the case of TLS version 1.2, it is important that mutual authentication of client and server be required. Supported cipher suites are identified by standards bodies. Authentication and authorization events can be logged and reported.

Implementations that support post-quantum cryptographic algorithms can be leveraged where practical.

8.3.3. Amplifier Management

The Amplifier information model does not include security functions. CableLabs specifications are available for HFC network elements that include security configuration and event monitoring and provide example implementations. Well-defined model-driven management does increase the attack surface of the amplifier; consequently, trusted models can be used and the Network Configuration Access Control Model [RFC 8341] can provide control of what clients can execute over RESTCONF.

The transponder and eAMP software can be updateable. The mechanism can implement secure update and secure software downloads available in CableLabs specifications.

8.3.4. Physical Security

Physical access to the Amplifier can be monitored and alarmed. Silicon devices used in the Amplifier, including management components and pluggable modules, can be tamper evident as well as tamper proof.

Hardware level interfaces such as JTAG and SPI are not recommended on production amplifiers.

Private keys, PKI certificates, or other credentials can be protected in tamper proof hardware and not be included in firmware or software.

9. Format and Content for Event, SYSLOG, and SNMP Notification

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

9.1. Event Notification

The Amplifier 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 where the Amplifier supports one event log (docsDevEventTable [RFC 4639]). Locally generated events are stored in the docsDevEventTable.
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.

Refer to [CCAP-YANG] for the DOCSIS CCAP events YANG module.

9.2. Amplifier Event Reporting

The Amplifier is required to log events and generate asynchronous notifications that indicate malfunction situations and notify the operator about important events.

The primary method for delivery of event reports is via RESTCONF messages. RESTCONF and YANG support defining asynchronous notification streams (like SNMP Traps). Notifications are subscribed to by the amplifier manager. The Amplifier asynchronously pushes notifications to any subscribed Amplifier Manager. YANG defines the notifications that can be subscribed to, as well as the payload to expect on that notification stream. This provides operational behavior where it is the responsibility of the Amplifier to notify the Amplifier Manager of events during run-time operation (via a push approach).

The Amplifier can be configured to send events directly to a Syslog server. This method allows an operator to collect detailed event and log details from the Amplifier without burdening the Amplifier Manager with receiving or reading these events from the Amplifier.

The Amplifier Manager maintains the responsibility of reporting events originating from an Amplifier by methods defined in this specification or through vendor specific methods such as the command line interface.

The Amplifier ***shall*** report all events for which the severity levels are enabled.

The Amplifier ***shall not*** report any event for which the severity level is disabled.

9.3. Format of Events

Section 9.5 lists events for the Amplifier.

This section describes how to report these events via the local event logging mechanism and Syslog.

Aside from the procedures defined in this document, event recording conforms to the requirements of [RFC 4639]. Event descriptions are defined in English.

9.3.1. Local Event Logging

The Amplifier ***shall*** maintain Local Log events in both local-volatile storage and local non- volatile storage.

The Amplifier *may* retain in local non-volatile storage events designated for local volatile storage.

The Amplifier *may* retain in local volatile storage events designated for local non-volatile storage.

The Amplifier ***shall*** implement its Local Log as a cyclic buffer with a minimum of ten entries.

The Amplifier Local Log non-volatile storage events ***shall*** persist across reboots.

The Amplifier ***shall*** persist undelivered event reports across reboots.

The Amplifier ***shall*** limit Event Message fields to no longer than 255 characters.

Events are considered identical if the EventId is the same and the event arguments are the same.

For identical events occurring consecutively, the Amplifier *may* choose to store only a single event.

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

The Amplifier ***shall*** clear both the local volatile and local non-volatile event logs when an event log reset is initiated through RESTCONF.

9.3.2. Standard Events

The EventId digit is a 32-bit unsigned integer. EventIds defined in [RFC 4639] ranging from 0 to (231 - 1) are reserved for DOCSIS technology.

The Amplifier ***shall*** report an EventId as a 32-bit unsigned integer and comply with the DOCSIS EventIds defined in Section 9.5.

[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 performing control functions or transmitting data, but which 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 control or data flow, but which will not cause the Amplifier to re-initialize. Error events could be reported in real time in either in an implementation specific manner or using RESTCONF.

Warning event (priority 5)

A failure occurred that could interrupt the normal control or data flow, but which will not cause the Amplifier to re-initialize. Warning events could be reported in real time in either in an implementation specific manner or using RESTCONF

Notice event (priority 6)

The event is important, but it is not a failure. Notice events could be reported in real time in either an implementation specific manner or using RESTCONF. 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. Informational events could be reported in real time in either in an implementation specific manner or using RESTCONF.

Debug event (priority 8)

Reserved for vendor-specific non-critical events.

During Amplifier initialization or reinitialization, the Amplifier ***shall*** persist event reporting configuration in its non-volatile memory.

After an Amplifier factory reset, the Amplifier ***shall*** support the default event reporting mechanism shown in Table 89.

The Amplifier reporting mechanism for each priority can be changed from the default reporting mechanism using remote configuration.

Table 89 – Amplifier Default Event Reporting Mechanism Versus Priority

Event Priority	Local Event Log	Notify
Emergency	Yes	No
Alert	Yes	No
Critical	Yes	No
Error	Yes	No
Warning	No	No
Notice	No	No
Informational	No	No
Debug	No	No

The Amplifier ***shall*** format notifications that it generates for standard DOCSIS events as specified in Section 9.5.

9.3.3. Vendor-Specific Events

An Amplifier ***shall*** 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

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 90 summarizes those considerations.

An Amplifier ***shall*** assign both DOCSIS events and vendor-specific events as indicated in Table 90.

Table 90 – Vendor-Specific Event Priorities Assignment

Event Priority	Amplifier Event Assignment
Emergency	Vendor-specific
Alert	Vendor-specific optional
Critical	Vendor-specific optional
Error	Vendor-specific optional
Warning	Vendor-specific optional
Notice	Vendor-specific optional
Informational	Vendor-specific optional
Debug	Vendor-specific optional

Vendor-specific optional event definitions are recommended only if the Amplifier allows for sufficient storage of such events.

9.3.4. Syslog

9.3.4.1. Syslog Support

The Amplifier learns of the Syslog server(s) IP addresses via DHCP during the address acquisition phase of initialization. Syslog messages are not encrypted and are available on the network in the clear although the Amplifier transponder may implement a mechanism to encrypt data transmissions.

The Amplifier ***shall*** support Syslog message transport compatible with [RFC 3164].

[RFC 3164] specifies a set of message severity levels. Note that the Syslog severity levels are zero-based whereas the DOCSIS-defined priority levels are one-based.

The Amplifier ***shall*** support Syslog remote server transport during boot and initialization.

The Amplifier ***shall*** use the remote Syslog servers from the DHCP options.

The Amplifier ***shall*** support Syslog remote server transport during operation only if the amplifier manager has configured remote Syslog servers.

The Amplifier ***shall not*** emit Syslog messages over Syslog remote server transport after initialization unless explicitly instructed to by an Amplifier Manager.

The Amplifier Manager *should* support configuring Syslog remote servers during Amplifier initialization for use by the Amplifier during operation. This configuration enables/disables Syslog remote server transport.

The Amplifier ***shall*** support up to two remote Syslog receivers.

The Amplifier ***shall*** provide Syslog messages to all Syslog receivers configured by DHCP (during boot/initialization) or the amplifier manager (during operation).

The Amplifier ***shall*** support configuration of IPv4 remote Syslog servers.

The Amplifier ***shall*** support configuration of IPv6 remote Syslog servers.

The Amplifier ***shall*** support filtering Syslog messages at a specified severity level.

During initialization, the Amplifier ***shall*** set this severity level to be warning (4) and above.

Filtered messages are not emitted to remote Syslog servers.

The Amplifier *may* locally log all Syslog messages regardless of configured severity level filters.

The Amplifier ***shall*** support configuration of the severity level filter after initialization.

The Amplifier ***shall*** support severity level filtering of Syslog messages provided over RESTCONF transport.

9.3.4.2. Syslog Throttling

The Amplifier ***shall*** support Syslog remote server transport throttling.

During boot and initialization, the Amplifier ***shall not*** apply any transport throttling.

During operation, the Amplifier ***shall*** support the configuration of a transport throttle.

The throttling can be configured to one of the following:

- unconstrained
- maintain below threshold
- stop at threshold
- inhibited.

If the Syslog transport throttle configuration is "unconstrained", the Amplifier does not apply transport throttling for that remote server.

If the Syslog transport throttle configuration is "maintain below threshold", the Amplifier ***shall*** throttle message transport when the threshold (messages per second) is reached.

The Amplifier ***shall*** resume unthrottled message transport when the message rate falls below the threshold.

If the Syslog transport throttle configuration is "stop at threshold", the Amplifier ***shall*** throttle message transport when the threshold (messages per second) is reached.

When set for "stop at threshold", the Amplifier ***shall not*** resume unthrottled message transport unless instructed.

The Amplifier ***shall*** emit a Syslog message to the remote servers before "stop at threshold" throttling occurs.

The Amplifier ***shall*** allow the "stop at threshold" threshold to be configurable.

If the Syslog transport throttle configuration is "inhibited", the Amplifier ***shall*** throttle all message transport to the remote server.

9.3.4.3. Syslog Message Format

The Amplifier sends Syslog messages compatible with [RFC 3164]. Although an updated version of the Syslog protocol was described in [RFC 5424], the latter format is incompatible with widely deployed log analyzers for the original Syslog protocol defined in [RFC 3164] and at the time of this specification is not widely supported in public domain source code libraries.

The required structure of the Syslog notification is a profile of [RFC 3164] and is defined as follows:

"<"+PRI+">"+TIMESTAMP+*space*+HOSTNAME+*space*+MSG+*space*+TAG

where:

PRI = an ASCII decimal value determined by multiplying the facility code by 8 and adding the Syslog priority level.

For kernel messages, the facility code = 0; for Amplifier "user" messages, the facility code = 1. Note that zero-based Syslog-Severity is used only for calculating PRI-VALUE. The zero-based Syslog-Priority is one less than the one-based DOCSIS-Priority value (i.e., Emergency = 0 and Debug = 7).

For example, for an Amplifier user message with a priority of Error: PRI = 1 * 8 + 3 = 11

Note that there is no space between the final ">" and the beginning of the TIMESTAMP. TIMESTAMP = "Mmm dd hh:mm:ss" (i.e., Jun 24 13:01:37)

HOSTNAME = Amplifier DeviceAlias. If the value of this attribute is not set, a vendor-specific value is provided, such as "NONE".

MSG = PROCESS+*space*+CONTENT+";"

PROCESS = alphanumeric string 1..32 chars with the process name terminated by the first non-alphanumeric character (e.g., a colon).

CONTENT = a string of printable characters; see [RFC 3164] for details on rules about the format of the CONTENT. For Amplifier events, the CONTENT includes the event number followed by the event message.

TAG = suffix composed of "AMPLIFIER-ID:"+*space*+UniqueId

UniqueId = a lower-case hexadecimal value (i.e., "12:34:56:78:ab:cd"), representing the MAC address of the Amplifier sending the Syslog notification. This is the same MAC Address used in the SystemStatus class.

Example:

<13>Jun 24 13:01:37 BSHPGATE06 dulcmgrd: Example message; AMPLIFIER-ID: 00:11:22:33:ab:cd

9.4. Clearing of Previously Reported Conditions

Certain events defined in Section 9.5 represent conditions that might exist for a specific duration while a condition exists. For example, Event ID 66070501 represents a normal operating temperature exceeded condition. An Amplifier might report a corresponding event defined which represents the state where the condition no longer exists or has been removed. In the example for Event ID 66070501, this condition is removed when the device's operating temperature falls within the normal operation parameters or temperature range.

Section 9.5 defines corresponding "clearing" events which can be used in these scenarios. In the example for Event ID 66070501, Event ID 66070511provides the ability for the Amplifier to clear the previously logged or alarmed condition. This event can be correlated to the previously logged or alarmed event through the parameter values carried in the event message. When the Amplifier reports clearing events is vendor specific.

9.5. Standard Events

Table 91 summarizes the format and content for event, syslog, SNMP notifications and NETCONF notifications required for the Amplifier. For additional information refer to [CM-OSSIv4.0] Annex D.

Each row specifies a possible event reported by the Amplifier. These events are to be reported by the Amplifier via one of the supported event mechanisms specified in Section 9.2.

The "Process" and "Sub-Process" columns indicate in which stage the event happens.

The "Priority" column indicates the priority the event is assigned in the Amplifier. These priorities are the same as is reported in the LEVEL field of the syslog.

The "Event Message" column specifies the event text, which is reported in the text field of the syslog and the notification message. The "Message Notes And Detail" column provides additional information about the event text in the "Event Message" column. Some of the text fields include variable information, which are often specified as key-value pairs. The variables are explained in the "Message Notes And Detail" column. For events where the "Event Message" or "Message Notes and Detail" column includes either <P1>, <P2> or <Pn>, there is a colon and single space between the value as defined by <P1>, <P2> or <Pn> and the preceding key text.

The key-value parameters are thus formatted as: [key]: [value]. Key value pairs are delimited by a semi-colon followed by a single space, as the following example indicates:

[key 1]: [value 1]; [key 2]: [value 2]; [key n]: [value n]

The "Event Message" field structure is defined as follows:

<Initial Event Message Text>; [key 1]: [value 1]; [key 2]: [value 2]; [key n]: [value n]; <TAGS>;

Keys which contain values which represent strings can enclose those strings within double quotations to prevent confusion if those string values contain a delimiter (colon or semicolon). Key strings should

capitalize each word (e.g., "Sensor Unit"). If a key's value is not present, the key is present, but the value is omitted (using a single space). For example:

key 1: ;

It is recommended that <Initial Event Message Text> string does not contain a semicolon since this is used as a delimiter. The <Initial Event Message Text> string follows a normal sentence capitalization scheme where the first word is capitalized as well as any defined terms and acronyms.

Refer to Annex D of [CM-OSSIv4.0] for the definition and format of "<TAGS>" keyword as part of the "Event Message" column for existing events defined in [CM-OSSIv4.0]. This specification will define the following convention for usage of the <TAGS> keyword as part of the "Event Message" column.

This specification defines the following keywords as part of the "Event Message" column:

"<TAGS>" (without the quotes) corresponds to: "<AMP-ID>" (without the quotes)

Where:

<AMP-ID>: eAMP MAC Address

Format*: "AMP-ID: xx:xx:xx:xx:xx:xx"

(*) without the quotes

The Amplifier ***shall*** support all applicable events defined in Annex D of [CM-OSSIv4.0].

The Amplifier ***shall*** support all applicable events defined in Annex B of [FMA-OSSI].

The Amplifier ***shall*** support all applicable events defined in Annex B of [RPHY-OSSI].

The Amplifier ***shall*** support all mandatory events as defined in Table 91.

The Amplifier *may* append additional vendor-specific text to the end of the event text reported in the syslog text field and the NETCONF notification message.

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 docsDevEvId object in [RFC 4639] or YANG message, and the <eventId> field of the syslog message.

Refer to [CANN] for the rules to generate unique Event IDs from the Error Code Set.

The "Notification Name" column specifies the SNMP or NETCONF notification, which notifies this event to a management notification receiver.

Table 91 – Amplifier Standard Events

Process	Sub-Process	Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
Init						
Init	Initialization	Notice	Local device initialization process passed; Descr: <P1>; <TAGS>;	P1 is optional P1 = Vendor Specific Event or Text	F700.01	70070001
Init	Reboot	Notice	Reboot; Type: <P1>; Reason: <P2>; Next SW Image Index: <P3>; <TAGS>;	P1 = {softReset, hardReset, nvReset, factoryReset} P2 = Text string indicating reason P3 = CurrentSwImageIndex of the Identification object after completing re-initialization	F700.02	70070002
Init	Operation	Notice	Move to operational successful; Amplifier Manager IP: <P1>; <TAGS>;	P1 = IP address of the Amplifier Manager which directed the move to operational	F700.03	70070003
Init	Initialization	Critical	Failure occurred during local device initialization process. Device reset; Descr: <P1>; <TAGS>;	P1 is optional P1 = Vendor Specific Event or Text	F700.04	70070004
Init	Config	Error	Unable to apply non-volatile configuration settings during local device initialization; Descr: <P1>; <TAGS>;	P1 is optional P1 = Vendor Specific Event or Text	F700.05	70070005
Init	Config	Error	Unable to apply default configuration settings during local device initialization; Descr: <P1>; <TAGS>;	P1 is optional P1 = Vendor Specific Event or Text	F700.06	70070006
Init	Config	Error	Unable to apply vendor-specific configuration settings during local device initialization; Descr: <P1>; <TAGS>;	P1 is optional P1 = Vendor Specific Event or Text	F700.07	70070007
Init	Config	Notice	Local device initialization completed without error; Descr:<P1>;<TAGS>;	P1 is optional P1 = Vendor Specific Event or Text	F700.43	70070043
Init	Announce	Notice	Announce connection start; Amplifier Manager IP: <P1>; <TAGS>;	P1: Amplifier Manager IP Address	F700.08	70070008
Init	Announce	Warning	Announce connection failure; Amplifier Manager IP: <P1>; Error: <P2>; <TAGS>;	P1: Amplifier Manager IP Address P2: Error Message	F700.09	70070009
Init	Announce	Notice	Announce redirect; Amplifier Manager IP: <P1>; <TAGS>;	P1: New Amplifier Manager IP Address	F700.10	70070010
Init	Announce	Notice	Announce success: Amplifier Manager IP: <P1>; <TAGS>;	P1: Amplifier Manager IP Address	F700.11	70070011
Init	Announce	Warning	Announce failure: Amplifier Manager IP: <P1>; Error: <P2>; <TAGS>;	P1: Amplifier Manager IP Address P2: Error Message	F700.12	70070012
Init	Announce	Warning	Timeout waiting for connection from Amplifier Manager; Amplifier Manager IP: <P1>; <TAGS>;	P1: Amplifier Manager IP Address	F700.13	70070013

Process	Sub-Process	Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
Init	DHCP	Critical	DHCP failed - Discover sent, no offer received; DHCP Server IP: <P1>; CIN Port: <P2>; <TAGS>;	P1 = DHCP server IP address P2 = CIN Port	F700.14	70070014
Init	DHCP	Critical	DHCP failed - Request sent, No response; DHCP Server IP: <P1>; CIN Port: <P2>; <TAGS>;	P1 = DHCP server IP address P2 = CIN Port	F700.15	70070015
Init	DHCP	Warning	DHCP WARNING - Non-critical field invalid in response; DHCP Server IP: <P1>; CIN Port: <P2>; <TAGS>;	P1 = DHCP server IP address P2 = CIN Port	F700.16	70070016
Init	DHCP	Critical	DHCP failed - Critical field invalid in response; DHCP Server IP: <P1>; CIN Port: <P2>; <TAGS>;	P1 = DHCP server IP address P2 = CIN Port	F700.17	70070017
Init	DHCP	Critical	DHCP failed - RS sent, no RA received; DHCP Server IP: <P1>; CIN Port: <P2>; <TAGS>;	P1 = DHCP server IP address P2 = CIN Port	F700.18	70070018
Init	DHCP	Critical	DHCP failed - Invalid RA; DHCP Server IP: <P1>; Port: <P2>; <TAGS>;	P1 = DHCP server IP address P2 = CIN Port	F700.19	70070019
Init	DHCP	Critical	DHCP failed - DHCP Solicit sent, No DHCP Advertise received; DHCP Server IP: <P1>; CIN Port: <P2>; <TAGS>;	P1 = DHCP server IP address P2 = CIN Port	F700.20	70070020
Init	DHCP	Critical	DHCP failed - DHCP Request sent, No DHCP REPLY received; DHCP Server IP: <P1>; CIN Port: <P2>; <TAGS>;	P1 = DHCP server IP address P2 = CIN Port	F700.21	70070021
Init	IPv6 Address Acquisition	Critical	Link-Local failure - Link-Local address failed DAD; DHCP Server IP: <P1>; CIN Port: <P2>; <TAGS>;	P1 = DHCP server IP address P2 = CIN Port	F700.22	70070022
Init	IPv6 Address Acquisition	Critical	DHCP lease failure - DHCP assigned address failed DAD; DHCP Server IP: <P1>; CIN Port: <P2>; <TAGS>;	P1 = DHCP server IP address P2 = CIN Port	F700.23	70070023
Init	IPv4 Address Acquisition	Notice	Successfully obtained IPv4 address; <TAGS>;		F700.24	70070024
Init	IPv6 Address Acquisition	Notice	Successfully obtained IPv6 address; <TAGS>;		F700.25	70070025
Init	ToD	Notice	Successfully obtained ToD; <TAGS>;		F704.02	70070402
Init	ToD	Error	ToD failed - ToD request sent - No response received; ToD Server IP: <P1>; Port: <P2>;<TAGS>;	P1 = ToD server IP address P2 = Ethernet port number	F704.03	70070403
Init	ToD	Error	ToD failed - Response received - Invalid data format; ToD Server IP: <P1>; Port: <P2>;<TAGS>;	P1 = ToD server IP address P2 = Ethernet port number	F704.04	70070404
Init	RESTCONF	Notice	RESTCONF initialization success; <TAGS>;		F700.35	70070035
Init	RESTCONF	Error	RESTCONF initialization failed: Reason:<P1>;<TAGS>	P1 = Text string indicating reason	F700.36	70070036

Process	Sub-Process	Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
Init	RESTCONF	Notice	RESTCONF connection accepted; Client IP:<P1>;<TAGS>;	P1 = Client IP address	F700.37	70070037
Init	RESTCONF	Warning	RESTCONF connection rejected; Client IP:<P1>; Reason: <P2>;<TAGS>;	P1 = Client IP address P2 = Text string indicating reason	F700.38	70070038
Init	RESTCONF	Notice	RESTCONF TLS connection success; Client IP:<P1>; <TAGS>;	P1 = Client IP address	F700.39	70070039
Init	RESTCONF	Warning	RESTCONF TLS connection failure; Client IP:<P1>; Reason:<P2>;<TAGS>;	P1 = Client IP address P2 = Text string indicating reason	F700.40	70070040
Init	RESTCONF	Notice	RESTCONF TLS connection closed gracefully; Client IP:<P1>;<TAGS>;	P1 = Client IP address	F700.41	70070041
Init	RESTCONF	Warning	RESTCONF TLS communication problem; Client IP:<P1>; Reason:<P2>; <TAGS>;	P1 = Client IP address P2 = Text string indicating reason	F700.42	70070042
Init	RESTCONF	Warning	RESTCONF request rejected; Client IP:<P1>; Reason:<P2>; <TAGS>;	P1 = MAC Manager IP address P2 = Text string indicating reason	F700.49	70070049
			Environmental			
Environmental	Temperature	Critical	High temperature threshold exceeded; Sensor Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = EntityIndex of temperature sensor P2 = Sensor reading (Celsius Temperature) P3 = value of CurrentDateTime at time of sensor reading	B705.00	66070500
Environmental	Temperature	Notice	High temperature threshold exceeded condition cleared; Sensor Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = EntityIndex of temperature sensor P2 = Sensor reading (Celsius Temperature) P3 = value of CurrentDateTime at time of sensor reading	B705.10	66070510
Environmental	Temperature	Warning	Normal operating temperature exceeded; Sensor Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = EntityIndex of temperature sensor P2 = Sensor reading (Celsius Temperature) P3 = value of CurrentDateTime at time of sensor reading	B705.01	66070501
Environmental	Temperature	Notice	Normal operating temperature exceeded condition cleared; Sensor Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = EntityIndex of temperature sensor P2 = Sensor reading (Celsius Temperature) P3 = value of CurrentDateTime at time of sensor reading	B705.11	66070511
			Physical			
Physical	Power	Critical	Excessive input current; Power Supply Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = EntityIndex of powerSupply P2 = Sensor reading (Amperes) P3 = value of CurrentDateTime at time of sensor reading	B705.02	66070502

Process	Sub-Process	Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
Physical	Power	Notice	Excessive input current condition cleared; Power Supply Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = EntityIndex of powerSupply P2 = Sensor reading (Amperes) P3 = value of CurrentDateTime at time of sensor reading	B705.12	66070512
Physical	Power	Critical	Improper input voltage; Power Supply Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = EntityIndex of powerSupply P2 = Sensor reading (Volts) P3 = value of CurrentDateTime at time of sensor reading	B705.03	66070503
Physical	Power	Notice	Improper input voltage condition cleared; Power Supply Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = EntityIndex of powerSupply P2 = Sensor reading (Volts) P3 = value of CurrentDateTime at time of sensor reading	B705.13	66070513
Physical	Power	Critical	Improper output voltage; Power Supply Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = Index of OutputRail P2 = Sensor reading (Volts) P3 = value of CurrentDateTime at time of sensor reading		
Physical	Power	Notice	Improper output voltage condition cleared; Power Supply Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = Index of OutputRail P2 = Sensor reading (Volts) P3 = value of CurrentDateTime at time of sensor reading		
Environmental	Security	Warning	Enclosure door opened; Time: <P1>; <TAGS>;	P1 = value of CurrentDateTime at time of sensor reading	B705.04	66070504
Environmental	Security	Notice	Enclosure door opened condition cleared; Time: <P1>; <TAGS>;	P1 = value of CurrentDateTime at time of sensor reading	B705.14	66070514
Environmental	Humidity	Error	High humidity threshold exceeded; Sensor Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = EntityIndex of humidity sensor P2 = Sensor reading (percent relative humidity) P3 = value of CurrentDateTime at time of sensor reading	B705.05	66070505
Environmental	Humidity	Notice	High humidity threshold exceeded condition cleared; Sensor Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = EntityIndex of humidity sensor P2 = Sensor reading (percent relative humidity) P3 = value of CurrentDateTime at time of sensor reading	B705.15	66070515
Environmental	Humidity	Warning	Normal operating humidity exceeded; Sensor Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = EntityIndex of humidity sensor P2 = Sensor reading (percent relative humidity) P3 = value of CurrentDateTime at time of sensor reading	B705.06	66070506

Process	Sub-Process	Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
Environmental	Humidity	Notice	Normal operating humidity exceeded condition cleared; Sensor Unit: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = EntityIndex of humidity sensor P2 = Sensor reading (percent relative humidity) P3 = value of CurrentDateTime at time of sensor reading	B705.16	66070516
Environmental	Security	Warning	Local management interface port activated; Port: <P1>; <TAGS>;	P1 = Management port identifier	B705.07	66070507
Environmental	Security	Notice	Local management interface port deactivated; Port: <P1>; <TAGS>;	P1 = Management port identifier	B705.17	66070517
Environmental	Security	Warning	Unsuccessful login on local management interface port; Port: <P1>; <TAGS>;	P1 = Management port identifier	B705.18	66070518
Environmental	Security	Notice	Successful login on local management interface port; Port: <P1>; <TAGS>;	P1 = Management port identifier	B705.19	66070519
Environmental	Sensor	Assigned by Vendor	Sensor event; Sensor Unit: <P1>; Type: <P2>; Scale: <P3>; Reading: <P4>; Descr: <P5>; <TAGS>;	P1 = EntityIndex of sensor P2 = SensorType P3 = Scale P4 = Sensor reading (Value) P5 = Entity Descr	B705.08	66070508
Environmental	Sensor	Notice	Sensor event condition cleared; Sensor Unit: <P1>; Type: <P2>; Scale: <P3>; Reading: <P4>; Descr: <P5>; <TAGS>;	P1 = EntityIndex of sensor P2 = SensorType P3 = Scale P4 = Sensor reading (Value) P5 = Entity Descr	B705.09	66070509
Environmental	Processor	Error	Processor overload threshold exceeded; Processor: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = Processor identifier (vendor specific) P2 = Processor utilization reading (percent) P3 = value of CurrentDateTime at time of detected overload condition	B705.20	66070520
Environmental	Processor	Notice	Processor overload threshold exceeded condition cleared; Processor: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = Processor identifier (vendor specific) P2 = Processor utilization reading (percent) P3 = value of CurrentDateTime at time of detected overload cleared condition	B705.21	66070521
Environmental	Storage	Error	Memory utilization threshold exceeded; Storage: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = HostResourcesStorage Index of local storage P2 = Memory utilization reading (vendor specific) P3 = value of CurrentDateTime at time of detected threshold exceeded condition	B705.22	66070522

Process	Sub-Process	Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
Environmental	Storage	Notice	Memory utilization threshold exceeded condition cleared; Storage: <P1>; Reading: <P2>; Time: <P3>; <TAGS>;	P1 = HostResourcesStorage Index of local storage P2 = Memory utilization reading (vendor specific) P3 = value of CurrentDateTime at time of detected threshold exceeded condition	B705.23	66070523
			Ingress Switch			
Amplifier	Ingress Switch	Notice	Ingress Switch state change. Now: <P1>; Was: <P2>; Attenuation: <P3>	P1 = current value of UsIngressSwitchCfg:State P2 = previous value of UsIngressSwitchCfg:State P3 = value of attenuation of dB in use. Can be zero.		
			AGC			
Amplifier	AGC	Notice	Downstream AGC enabled. DsAgcEnabled:<P1>; DsAgcPilotLossProtection type:<P2>; Port:<P3>	P1 = current value of DsAgcEnabled P2 = current value of DsAgcPilotLossProtection P3 = RfPortIndex		
Amplifier	AGC	Notice	Downstream AGC disabled. DsAgcEnabled:<P1>; Port:<P2>	P1 = current value of DsAgcEnabled P2 = RfPortIndex		
Amplifier	AGC	Warning	Downstream AGC pilot lost. Port:<P1>; Pilot Number:<P2>; Pilot Frequency:<P3>	P1 = RfPortIndex P2 = PilotNum P3 = PilotFreq		
Amplifier	AGC	Notice	Downstream AGC pilot loss cleared. Port:<P1>; Pilot Number:<P2>; Pilot Frequency:<P3>	P1 = RfPortIndex P2 = PilotNum P3 = PilotFreq		
Amplifier	AGC	Critical	Downstream AGC too close to rail. Port:<P1>; DsAgcToUpperLimit:<P2>; DsAgcToLowerLimit:<P3>	P1 = RfPortIndex P2 = DsAgcToUpperLimit P3 = DsAgcToLowerLimit		
Amplifier	AGC	Notice	Downstream AGC too close to rail condition cleared. Port:<P1>; DsAgcToUpperLimit:<P2>; DsAgcToLowerLimit:<P3>	P1 = RfPortIndex P2 = DsAgcToUpperLimit P3 = DsAgcToLowerLimit		
			Data Collection			
Data Collection	Data File	Warning	Bulk data file size reached high threshold. Data collection type: <P1>; Filename: <P2>; File maximum size: <P3>; <TAGS>;	P1 = (Type of data collection, e.g., debug) P2 = Filename assigned to the data file P3 = maximum size of capture file in bytes	F007.01	70000701
Data Collection	Data File	Warning	Bulk data file size reached full threshold. Data collection type: <P1>; Filename: <P2>; File maximum size: <P3>; <TAGS>;	P1 = (Type of data collection, e.g., debug) P2 = Filename assigned to the data file P3 = maximum size of capture file in bytes	F007.02	70000702
Data Collection	PNM test	Informational	PNM test complete; Test Id: <P1>; Test Type: <P2>; Test Status: <P3>; <TAGS>;	P1 = PNM Test Id P2 = PNM Test Type P3 = PNM Test Status	F007.03	70000703

Process	Sub-Process	Priority	Event Message	Message Notes and Detail	Error Code Set	Event ID
Data Collection	Data File	Informational	Bulk data file status update; FileType: <P1>; Filename: <P2>; File Status: <P3>; <TAGS>;	P1 = FileType::FileType P2 = Filename::LocalFilename P3 =FileStatus::FileStatus	F007.04	70000704