PacketCable™ Specification

Multimedia Specification

PKT-MM-I09-230913

ISSUED

DISCLAIMER

Document Status Sheet

Document Control Number:	PKT-MM-I09-230913
Document Title:	Multimedia Specification
Revision History:	I01 - Released 06/27/03 I02 - Released 09/30/04 I03 - Released 12/21/05 I04 - Released 05/22/08 I05 - Released 10/29/09 I06 - Released 06/29/11 I07 - Released 11/11/15 C01 - Closed 11/20/19 I08 - Released 11/04/22 I09 - Released 09/13/23
Date:	September 13, 2023
Status:	~~Work in Progress~~ ~~Draft~~ **Issued** ~~Closed~~
Distribution Restrictions:	~~Author Only~~ ~~CL/Member~~ ~~CL/ Member/ Vendor~~ **Public**

Key to Document Status Codes:

Work in Progress An incomplete document designed to guide discussion and generate feedback; may include several alternative requirements for consideration.

Draft A document that is considered largely complete but is undergoing review by members and vendors. Drafts are susceptible to substantial change during the review process.

Issued A public document that has undergone rigorous member and vendor review, supports cross-vendor interoperability, and is suitable for certification/qualification testing. Issued specifications are subject to the Engineering Change process.

Closed A static document that has been reviewed, tested, validated, and closed to further Engineering Change requests to the specification through CableLabs.

Trademarks

Contents

List of Figures

List of Tables

1 INTRODUCTION

1.1 Purpose

The intent of this specification is to support the deployment of general Multimedia services by providing a technical definition of several IP-based signaling interfaces that leverage core QoS and policy management capabilities native to DOCSIS® Versions 1.1 and greater. Throughout this specification, the term DOCSIS will be used to refer to any DOCSIS version greater than or equal to 1.1. For the purposes of this specification, Multimedia services may be defined as IP-based services (e.g., online gaming, video-conferencing, streaming media, etc.) requiring QoS-based network resources (as contrasted with services such as web browsing, e-mail, instant messaging and file-sharing that are commonly provided using best-effort flows). While telephony or voice-based services are not specifically excluded from this definition, the PacketCable 1.x set of specifications provide coverage specific to this type of service delivery, and, therefore, those specifications should be consulted as appropriate.

PacketCable Multimedia defines a service delivery framework that provides general-purpose QoS, event-based accounting, and security functionality founded upon the mechanisms defined in PacketCable 1.x. However, due to the broader spectrum of applications and services addressed by this initiative, each of these functional areas has been revisited and generalized for the present purposes. Telephony-specific requirements and interfaces (e.g., call signaling, PSTN interconnection and electronic surveillance) are not part of PacketCable Multimedia, while core functionality such as QoS resource management mechanisms, has been enhanced. Throughout this process, one of the primary objectives of this work has been to leverage and reuse as much of the existing body of PacketCable 1.x investment, knowledge base, and technical functionality as possible.

Key features of the described Multimedia service delivery framework include:

- Simple, powerful access to DOCSIS QoS mechanisms supporting both time and volume-based network resource authorizations,

- Abstract, event-based network resource auditing and management mechanisms,

- A robust security infrastructure that provides integrity and appropriate levels of protection across all interfaces.

1.2 Scope

As outlined in the accompanying technical report [MM-ARCH], the current scope of this specification is limited to network-based QoS resource management and usage auditing capabilities. This approach was motivated by several criteria, including rapid time-to-market for QoS-enhanced Multimedia services (which may take the form of new applications or existing applications retrofitted per this specification), the absence of QoS signaling requirements on CPE devices, and security assurances provided by an absence of client-based QoS signaling.

In addition, the QoS signaling mechanisms outlined herein provide the foundation for client-based QoS signaling mechanisms which may be defined in the future. As such scenarios have already been considered and documented in the accompanying technical report, one of the secondary goals of this version of this document is to provide for smooth migration to a client-based model as market and technology advances dictate. It should also be noted that the scope of the QoS framework defined in this specification is limited to the access network and that the choice of specific backbone QoS strategies is left as an administrative decision to be addressed by each service provider as it deems best.

Consistent with all CableLabs specifications, requirements defined within this document will address multi-vendor interface compatibility and interoperability. Specific performance and fault-tolerance strategies are left as vendor differentiators.

1.3 Relation to PacketCable 1.x

Since its inception the PacketCable initiative has been positioned as an IP-based Multimedia service deployment infrastructure, exploiting and enhancing the underlying QoS capabilities of the DOCSIS access network. Based on market demand and technology readiness, VoIP was chosen as the first IP-based service to leverage these unique

broadband access network capabilities, as the PacketCable 1.x suite of specifications simultaneously defines both a general QoS-based service delivery framework and a number of telephony-specific functional elements and mechanisms.

Specifically, the PacketCable 1.x VoIP architecture provides specifications promoting multi-vendor interoperability and CableLabs certification and qualification in the following key telephony areas:

- Network-based call signaling (NCS)

- Inter-domain call signaling (CMSS)

- PSTN gateway and interop functionality (TGCP, ISTP)

- Access-network and backbone QoS capabilities (DQoS, IQoS)

- Event messaging with telephony-specific extensions (EM)

- CPE endpoint provisioning and monitoring (Provisioning and SNMP MIBs)

- Comprehensive suite of interface security mechanisms (Security)

- Primary-line telephony characteristics

- Electronic surveillance

Since the intent of this specification is to extract the functional core of this architecture in support of the enhancement and deployment of other Multimedia services, support for these telephony-specific features is not required, while the core QoS, event messaging and security mechanisms are emphasized and generalized. The result is a framework that provides IP-based access to DOCSIS access-network QoS mechanisms complemented by secure and robust authorization and audit mechanisms.

In keeping with the PacketCable 1.x Dynamic Quality of Service (DQoS) model, the primary logical construct in support of QoS-based service delivery is a Gate (along with its accompanying authorization token). A PacketCable Gate represents a policy-based authorization for a specific envelope of network resources characterized by a suite of QoS parameters, as well as classifiers for originating and terminating IP addresses and ports, which define and limit the scope of the associated QoS-enhanced flow.

PacketCable 1.x defines a pre-authorization model in which Gates are created and installed at policy enforcement points (e.g., a CMTS) prior to actual network resource reservation or activation requests. This process, termed Gate Control, is managed through a COPS-based policy interface on the CMTS. In this specification, this interface is generalized and enhanced to allow for full lifecycle management of QoS-based service flows through this policy interface. That is, Gate installation and management is supplemented with service flow creation, modification and deletion functions to provide for network-based QoS service delivery.

To complement these enhanced QoS policy and signaling capabilities, the RADIUS-based event messaging infrastructure of PacketCable 1.x is carried over, but with telephony-specific primitives tagged as optional. Since the EM protocol was designed with a considerable amount of generality and abstraction, this approach provides continuity with existing protocol and Record Keeping Server (RKS) implementation while still allowing Multimedia vendors to select from a set of supplementary protocol primitives on an as-needed basis. For example, if a QoS-based video conferencing service were being developed based on this specification, a number of telephony-specific EM message types (e.g., Call_Answer and Call_Disconnect) may map directly to the service's audit requirements.

Finally, in order to provide robust security for this Multimedia framework, IPsec will be applied to the COPS and RADIUS interfaces in a manner analogous to the corresponding interfaces in the PacketCable 1.x architecture.

1.4   Specification Language

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

| "MUST" | This word means that the item is an absolute requirement of this specification. |
| "MUST NOT" | This phrase means that the item is an absolute prohibition of this specification. |

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

2   REFERENCES

2.1   Normative References

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

[DOCSIS MULPIv3.0] DOCSIS 3.0 MAC and Upper Layer Protocols Interface Specification, CM-SP-MULPIv3.0-C01-171207, December 7, 2017, Cable Television Laboratories, Inc.

[DOCSIS MULPIv3.1] DOCSIS 3.1 MAC and Upper Layer Protocols Interface Specification, CM-SP-MULPIv3.1-I25-230419, April 19, 2023, Cable Television Laboratories, Inc.

[DOCSIS MULPIv4.0] DOCSIS 4.0 MAC and Upper Layer Protocols Interface Specification, CM-SP-MULPIv4.0-I07, 230503, May 3, 2023, Cable Television Laboratories, Inc.

[IETF RFC 1305] Network Time Protocol (Version 3) Specification, Implementation and Analysis, March 1992.

[IETF RFC 2205] Resource ReSerVation Protocol (RSVP)—Version 1 Functional Specification, September 1997.

[IETF RFC 2210] The Use of RSVP with IETF Integrated Services, September 1997.

[IETF RFC 2211] Specification of the Controlled-Load Network Element Service, September 1997.

[IETF RFC 2212] Specification of Guaranteed Quality of Service, September 1997.

[IETF RFC 2216] Network Element Service Specification Template, September 1997.

[IETF RFC 2474] Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers, December 1998.

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

[IETF RFC 2748] The COPS (Common Open Policy Service) Protocol, January 2000.

[IETF RFC 2753] A Framework for Policy-based Admission Control, January 2000.

[IETF RFC 2866] RADIUS Accounting, June 2000.

[IETF RFC 3084] COPS Usage for Policy Provisioning (COPS-PR), March 2001.

[PKT-DQOS1.5] PacketCable 1.5 Dynamic Quality of Service, PKT-SP-DQOS1.5-C01-191120, November 20, 2019, Cable Television Laboratories, Inc.

[PKT-EM1.5] PacketCable 1.5 Event Messages, PKT-SP-EM1.5-C01-191120, November 20, 2019, Cable Television Laboratories, Inc.

[MM-ARCH] PacketCable Multimedia Architecture Framework, PKT-TR-MM-ARCH-C01-191120, November 20, 2019, Cable Television Laboratories, Inc.

[PKT-SEC1.5] PacketCable 1.5 Security, PKT-SP-SEC1.5-C01-191120, November 20, 2019, Cable Television Laboratories, Inc.

[BPI+] DOCSIS Baseline Privacy Plus Interface Specification, CM-SP-BPI+-C01-081104, November 4, 2008, Cable Television Laboratories, Inc.

2.2   Informative References

This specification uses the following informative references.

[CCAP-OSSIv3.1] DOCSIS 3.1 CCAP OSSI Specification, CM-SP-CCAP-OSSIv3.1-I26-230613, Cable Television Laboratories, Inc.

[IETF RFC 2751] S. Herzog. Signaled Preemption Priority Policy Element, January 2000.

[MM-WS] PacketCable Multimedia Web Service Interface Specification, PKT-SP-MM-WS-C01-191120, November 20, 2019, Cable Television Laboratories, Inc.

[IETF RFC 3168] The Addition of Explicit Congestion Notification (ECN) to IP, September 2001.

[IETF RFC 3171] IANA Guidelines for IPv4 Multicast Address Assignments, March 2010.

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

2.3   Reference Acquisition

- Cable Television Laboratories, Inc., 858 Coal Creek Circle, Louisville, CO 80027; Phone +1-303-661-9100; Fax +1-303-661-9199; Internet: www.cablelabs.com, www.packetcable.com

- Internet Engineering Task Force (IETF) Secretariat, 48377 Fremont Blvd., Suite 117, Fremont, California 94538, USA, Phone: +1-510-492-4080, Fax: +1-510-492-4001, http://www.ietf.org

3   TERMS AND DEFINITIONS

This specification uses the following terms.

Access Control	Limiting the flow of information from the resources of a system only to authorized persons, programs, processes, or other system resources on a network.
Active	A service flow is said to be "active" when it is permitted to forward data packets. A service flow must first be admitted before it is active.
Admitted	A service flow is said to be "admitted" when the CMTS has reserved resources (e.g., bandwidth) for it on the DOCSIS network.
Authentication	The process of verifying the claimed identity of an entity to another entity.
Authorization	The act of giving access to a service or device if one has permission to have the access.
DiffServ	Differentiated Services (RFC2475) refers to an architectural approach to providing QoS in which packets are tagged with a priority designation associated with various service levels on the network.
Downstream	The direction from the head-end toward the subscriber location.
Event Message	A message capturing a single step during the lifetime of a usage session. In this report event messages generally reference policy decisions or QoS changes.
Flow [DOCSIS Flow]	A unidirectional sequence of packets associated with a Service ID and a QoS. Multiple Multimedia streams may be carried in a single DOCSIS Flow. Also known as a DOCSIS "service flow"
Flow [IP Flow]	A unidirectional sequence of packets identified by OSI Layer 3 and Layer 4 header information. This information includes source/destination IP addresses, source/destination port numbers, protocol ID. Multiple Multimedia streams may be carried in a single IP Flow.
IntServ	Integrated Services (RFC1633) refers to an architectural approach to providing QoS in which multiple flows requiring real-time and non-real-time treatment are mixed over shared network links. Resources are reserved and state is maintained on a per-flow basis.
Originating Leg	Segment of a service session geographically associated with originating endpoint (i.e., calling party in the telephony model).
Terminating Leg	Segment of a service session geographically associated with the terminating endpoint (i.e., called party in the telephony model).
Upstream	The direction from the subscriber location toward the headend.

4   ABBREVIATIONS AND ACRONYMS

This specification uses the following abbreviations.

AM	Application Manager. A system that interfaces to Policy Server(s) for requesting QoS-based service on behalf of an end-user or network management system.
AS	Application Server. A server that interfaces to the PacketCable Multimedia Application Manager that requests PacketCable Multimedia services on behalf of clients.
BCID	Billing Correlation ID. Defined in the PacketCable Event Messaging specification.
CM	DOCSIS® Cable Modem.
CMS	Call Management Server. Network element defined in the PacketCable 1.0 Architecture technical report.
CMTS	Cable Modem Termination System. Device at a cable head-end which implements the DOCSIS RFI MAC protocol and connects to CMs over an HFC network.
COPS	Common Open Policy Service. Defined in RFC2748.
DOCSIS	Data-Over-Cable Service Interface Specifications. A set of interface specifications for transmitting data over cable television systems in a standard fashion.
DQoS	Dynamic Quality-of-Service.
DSx (Messaging)	DOCSIS QoS signaling mechanism providing Dynamic Service Add, Change and Delete semantics.
FQDN	Fully Qualified Domain Name.
HFC	Hybrid Fiber/Coax. An HFC system is a broadband bi-directional shared media transmission system using fiber trunks between the head-end and the fiber nodes, and coaxial distribution from the fiber nodes to the customer locations.
IETF	Internet Engineering Task Force. A body responsible for, among other things, developing standards used on the Internet. (See http://www.ietf.org.)
IGMP	Internet Group Management Protocol
IP	Internet Protocol.
KDC	Key Distribution Center. Network element defined in the PacketCable 1.0 Architecture technical report. Also defined in the PacketCable Security specification.
MG	Media Gateway. Network element defined in the PacketCable 1.0 Architecture technical report.
MGC	Media Gateway Controller. Network element defined in the PacketCable 1.0 Architecture technical report.
MLD	Multicast Listener Discovery
MSO	Multiple Service Operator.
MTA	Multimedia Terminal Adapter. Network element defined in the PacketCable 1.0 Architecture technical report. Contains the interface to a physical voice device, a network interface, CODECs, and all signaling and encapsulation functions required for VoIP transport, class features signaling, and QoS signaling. May be implemented as a standalone (S-MTA) or embedded (E-MTA) device, depending upon whether a DOCSIS CM is integrated.
NAT	Network Address Translation. Function performed to convert IP addresses, and potentially transport ports (PAT), from one network address numbering convention to another.
PDP	Policy Decision Point. Defined in RFC2753.
PEP	Policy Enforcement Point. Defined in RFC2753.
PS	Policy Server. A system that primarily acts as an intermediary between Application Manager(s) and CMTS(s). It applies network policies to Application Manager requests and proxies messages between the Application Manager and CMTS.
PSTN	Public Switch Telephone Network.
QoS	Quality of Service. Method used to reserve network resources and guarantee availability for applications.

RADIUS	Remote Authentication Dial-In User Service. Defined in RFC2138 and RFC2139. An Internet protocol originally designed for allowing users dial-in access to the Internet through remote servers. Its flexible design has allowed it to be extended well beyond its original intended use.
RAP	Resource Allocation Protocol Working Group in the IETF. Responsible for the definition an maintenance of the COPS protocol.
RCD	Resource Control Domain. A logical grouping of elements that provide connectivity and network resource level policy management along the packet forwarding paths to and from an end host. The RCD consists of CMTS and Policy Server entities whose responsibilities include management of resources along the packet forwarding paths.
RFC	Request for Comments. Technical policy documents approved by the IETF which are available at http://www.ietf.org/rfc.html
RFI	DOCSIS Radio Frequency Interface specification, defining MAC and Physical layer interfaces between CMTS and CM network elements.
RKS	Record Keeping Server. Network element defined in PacketCable 1.0 Architecture technical report. Also defined in the PacketCable Event Messaging specification. The device which collects and correlates the various Event Messages.
RSVP	Resource Reservation Protocol. Defined in RFC2205.
RSVP+	PacketCable profile and extension of RSVP, defined in the PacketCable DQoS specification.
SCD	Session Control Domain. A logical grouping of elements that offer applications and content to service subscribers. The Application Manager resides in the SCD.
S-MTA	Standalone MTA. A single node that contains an MTA and a non-DOCSIS MAC (e.g., Ethernet).
TCP	Transmission Control Protocol.
TLV	Type-Length-Value. Technique used in formatting protocol elements.
UDP	User Datagram Protocol. A connectionless protocol built upon Internet Protocol (IP).
UGS	Unsolicited Grant Service. DOCSIS QoS scheduling type used for constant bit rate services (e.g., voice codecs).
UGS/AD	Unsolicited Grant Service with Activity Detection. DOCSIS QoS scheduling type used for constant bit rate services with periodic inactivity (e.g., voice codecs implementing silence suppression).
VoIP	Voice over IP.
VPN	Virtual Private Network.

5 TECHNICAL OVERVIEW

This section consists of background material which some readers may find to be useful context for the detailed protocol interface specifications that follow. The intent in this section is to provide a high-level overview of the PacketCable Multimedia architecture and the fundamental technologies upon which it is based.

5.1 QoS Background

As noted throughout this specification, one of the primary features of the PacketCable Multimedia service framework lies in the fact that it provides IP-layer access to sophisticated QoS capabilities defined in DOCSIS versions 1.1 and greater and PacketCable 1.x. This subsection provides a brief overview of these capabilities as preparatory background for the detailed QoS policy and resource management discussion that follows.

5.1.1 DOCSIS QoS Summary

The DOCSIS 3.0 MAC and Upper Layer Protocols Interface Specification [DOCSIS MULPIv3.0] defines a set of QoS facilities based on a fundamental network resource management construct known as a Service Flow. A Service Flow is defined as "a MAC-layer transport service which (1) provides unidirectional transport of packets from the upper layer service entity to the RF, and (2) shapes, polices, and prioritizes traffic according to QoS traffic parameters defined for the flow." In addition to this primary abstraction facilitating the reservation and scheduling of shared access network resources on a per-flow basis, a number of tangible supporting constructs are defined and used to manage these resources. Three of these are:

Service Flow Encodings: type-length-value (TLV) encoded parameters used to define QoS parameters associated with a Service Flow.

Classifier: type-length-value (TLV) encoded IP, Ethernet and IEEE802.1p/q parameters used to define and limit the scope of a flow in terms of originating and terminating endpoints.

Payload Header Suppression Rules: type-length-value (TLV) encoded parameters used by the sending entity to suppress repetitive payload headers following the DOCSIS Extended Header field and used by the receiving entity to restore the suppressed header information. PacketCable Multimedia does not support this functionality at this time.

While DOCSIS supports provisioned (i.e., static, long-lived Service Flows that are established during the CM registration process) and dynamic (i.e., transient Service Flows that are added, modified and deleted on an as-needed basis) QoS models, the PacketCable Multimedia framework is primarily concerned with the dynamic variety as this allows for optimal network resource management through statistical multiplexing as service requirements demand.

Service Flow management is performed through MAC-layer DOCSIS Dynamic Service Add/Change/Delete (DSA/DSC/DSD) messaging, which may be initiated either by the CM or by the CMTS. DSA and DSC transactions take the form of a three-way exchange in which a request (REQ) is followed by a response (RSP) which is then acknowledged (ACK). DSD messages are simple two-way exchanges. A specific attribute known as a Confirmation Code is provided in each DSx response message and indicates the success or failure status of a transaction.

One important point to note in reviewing the QoS capabilities provided in DOCSIS is that upstream and downstream Service Flows receive fundamentally different treatment at the CMTS. This is a result of the fact that upstream RF channels are contentious, shared-access mediums, taking the topological form of a many-to-one relationship between multiple CMs and a single CMTS. Conversely, the downstream RF channel behaves much more akin to a traditional IP router in which packets arrive (either from the access network or over backbone trunks), are queued, and are forwarded to one or more destinations. Consequently, distinct QoS mechanisms are applied depending upon whether a particular uni-directional Service Flow is oriented in the upstream or downstream direction.

Upstream Service Flows may be defined with one of five service flow scheduling types:

Best-Effort: a standard contention-based resource management strategy in which transmit opportunities are granted on a first-come-first-served basis, albeit under the coordination of the CMTS scheduler. This scheduling type may be supplemented with QoS characteristics in which, for example, maximum rate limits are applied to a particular Service Flow.

Non-Real-Time Polling: a reservation-based resource management strategy in which a particular CM is polled on a fixed interval to determine whether data has been queued for transmission on a particular service flow, and, if so, a transmission opportunity or grant for that service flow is provided by the scheduler.

Real-Time Polling: analogous to the Non-Real-Time-Polling scheduling type, except that the fixed polling interval is typically very short (<500ms). Polling scheduling types are most suitable for variable- bit-rate traffic that has inflexible latency and throughput requirements.

Unsolicited Grant: a reservation-based resource management strategy in which a fixed-size grant is provided to a particular Service Flow at (nearly) fixed intervals without additional polling or interaction. This scheduling type is most suitable for constant bit-rate traffic and eliminates much of the protocol overhead associated with the polling types.

Unsolicited Grant with Activity Detection: a reservation-based resource management strategy that represents a hybrid of the polling and unsolicited grant scheduling types in which fixed grants are provided at (nearly) fixed intervals so long as data is queued for transmission. During periods of inactivity, this scheduling type reverts to a polling mode in order to conserve unused bandwidth.

Due to the unique nature and specialized characteristics of each of these scheduling types, specific QoS parameters are associated with each. These parameters are outlined in detail in the section that follows.

Downstream Service Flows are defined using the same set of QoS parameters that are associated with the Best-Effort scheduling type on the upstream.

Regardless of the orientation of the flow or the particular scheduling type requested, all dynamic Service Flows procedure through three logical states, summarized below. While certain optimized signaling scenarios allow for a so-called "single-phase" commit operation, the request still proceeds logically through all three phases as it is serviced at the CMTS.

Authorized: requests are authenticated and network policy rules are applied resulting in an authorization envelop forming the boundary of subsequent reservation requests.

Admitted (or Reserved): an inactive Service Flow is constructed and resources are reserved by the scheduler so that subsequent activation requests are guaranteed to succeed; reserved resources may be used by best-effort traffic (from the same or different CMs) until committed.

Active (or Committed): the Service Flow is activated, along with corresponding Classifiers; QoS-enhanced packets are now able to traverse the flow.

Note: In a literal sense, DOCSIS does not define "states", but, rather, "attributes" of Service Flows which are completely replaced with each DSC transaction. The states described here are a logical construct used in a conceptual model describing the resource management process performed on the CMTS. Also, the DOCSIS RFI specification standardizes upon the terms "admitted" and "active" in defining the attributes of a service flow, while PacketCable has adopted the equivalent terms "reserved" and "committed" in characterizing Gate states.

While DOCSIS does not define a specific authorization procedure to be applied to DSx messages, it does provide protocol support through a facility known as an Authorization Block for service-specific authorization schemes. Any credentials or authorization tokens that are presented via the Authorization Block are forwarded to an appropriate authorization module prior to the processing of the DSx request on the CMTS. PacketCable makes extensive use of this authorization mechanism as described below.

5.1.1.1   DOCSIS 3.0 Multicast QoS Framework

In DOCSIS 3.0, a framework was developed for providing special QoS treatment for IP Multicast Sessions that were signaled via IP Multicast protocols such as IGMP and MLD. This framework relies on an incoming membership request or "JoinMulticastSession" to be evaluated against the CmtsGroupConfig table for the purposes of determining if special QoS handling needed for a specific multicast session.

This specification allows a Gate-Set message be treated as the functional equivalent of a "JoinMulticastSession". This specification replaces the use of the configured CmtsGroupConfigTable with dynamic PacketCable Multimedia based Gate-Set messages for configuring the QoS of IP multicast replications to subscribers. Multicast Gates are expected to use application level encryption as PacketCable Multimedia does not define encryption of IP multicast

flows. Furthermore for Multicast Gates, PacketCable Multimedia overrides the use of the IP Multicast Join Authorization feature defined in [DOCSIS MULPIv3.0].

Addition of non-PacketCable Multimedia signaled IP Multicast Joiners to an existing PacketCable Multimedia-initiated Group Service Flow (GSF) [DOCSIS MULPIv3.0] is beyond the scope of this specification. The combination of non-PacketCable Multimedia signaled IP Multicast Joiners and PacketCable Multimedia-initiated Multicast joiners onto the same GSF is handled in a vendor specific manner.

An operator should configure the IP Multicast Join Authorization feature to reject unauthorized attempts by non-PacketCable Multimedia authorized Multicast Joins to join multicast sessions intended to be authorized only by PacketCable Multimedia.

PacketCable Multimedia Multicast Gates are implemented using DOCSIS Single-Session Group Service Flows.

The operation of PacketCable Multimedia Multicast Gates for subscribers reached through pre-3.0 CMs is beyond the scope of this specification.

5.1.2   PacketCable 1.x QoS Summary

While the DOCSIS 3.0 MAC and Upper Layer Protocols Interface [DOCSIS MULPIv3.0] specification defines the fundamental QoS mechanisms that form the core of the PacketCable DQoS model, the PacketCable DQoS specification [PKT-DQOS1.5] augments these capabilities with a COPS-based policy management framework. Just as the Service Flow represents the primary abstraction in the DOCSIS QoS model, the Gate plays a comparably significant role in the PacketCable DQoS scheme. A Gate defines a resource authorization envelope consisting of IP-level QoS parameters as well as classifiers defining the scope of Service Flows that may be established against the Gate. In accordance with the DOCSIS authorization mechanisms described above, only DSx requests which conform to the following general relation on a parameter-by-parameter basis will be granted:

$$\text{Authorized Envelope} \geq \text{Reserved Envelope} \geq \text{Committed Envelope}$$

Based in this policy management model, PacketCable 1.x defines a pre-authorization scheme in which network resources are authorized in advance of DSx messaging that requests establishment of a corresponding Service Flow. Consequently, the COPS interface used to install and manage Gates corresponds more closely with the COPS-PR model defined in [IETF RFC 3084] than with the standard COPS scheme specified in [IETF RFC 2748]. Also, in order to install and manage these Gates, the PacketCable DQoS specification defines a set of COPS client-specific objects which constitute the primitives of a Gate Control signaling interface between the CMS and the CMTS.

Specifically, the CMS may be logically decomposed into a Call Agent, responsible for telephony call-state maintenance, and a Gate Controller, which receives authorization requests from the Call Agent (through an internal interface) and installs policy decisions in the form of Gates on the CMTS. In the PacketCable Multimedia model, this decomposition is formalized through two separate network elements, the Policy Server (analogous to the PacketCable 1.x Gate Controller) and the Application Manager (defining service-specific functionality similar to the Call Agent in the PacketCable 1.x model).

As an illustration of this pre-authorization model and the use of the Gate Control interface on the CMTS, a typical single-zone (i.e., using a single CMS) on-net PacketCable 1.x call flow proceeds as follows (some of these steps would normally happen in parallel):

- E-MTA$_o$ boots, provisions and registers with CMS

- CMS issues request to E-MTA$_o$ for notification of off-hook event and dialed-digits

- E-MTA$_t$ boots, provisions and registers with CMS

- CMS issues request to E-MTA$_t$ for notification of off-hook event and dialed-digits

- E-MTA$_o$ goes off-hook, notifies CMS and delivers dialed-digits

- CMS issues a request to E-MTA$_o$ to create a new logical connection and retrieves SDP$_o$

- CMS issues a request to E-MTA$_t$ to create a new logical connection and retrieves SDP$_t$

- CMS installs a Gate on CMTS$_o$ and retrieves corresponding GateID$_o$ token

- CMS installs a Gate on $CMTS_t$ and retrieves corresponding $GateID_t$ token

- CMS issues a request (with $GateID_o$) to $E-MTA_o$ to reserve resources and play ringback

- $E-MTA_o$ issues a DSA-REQ to $CMTS_o$ to establish service flows and reserve resources

- CMS issues a request (with $GateID_t$) to $E-MTA_t$ to reserve resources and play an alerting tone

- $E-MTA_t$ issues a DSA-REQ to $CMTS_t$ to establish service flows and reserve resources

- $E-MTA_t$ goes off-hook and notifies CMS

- CMS issues a request to $E-MTA_o$ to stop playing ringback, commit resources and cut-through media path

- $E-MTA_o$ issues a DSC-REQ to $CMTS_o$ to commit resources

- CMS issues a request to $E-MTA_t$ to commit resources and cut-through media path

- $E-MTA_t$ issues a DSC-REQ to $CMTS_t$ to commit resources

- Call proceeds

In contrast to the PacketCable 1.x model in which the client device (i.e., E-MTA) initiates the resource reservation and activation procedures, the PacketCable Multimedia resource management model allows for the proxying of these steps on behalf of the endpoint through an enhanced Gate Control interface.

This concludes this brief review of DOCSIS and PacketCable 1.x QoS fundamentals. For further details related to either of these considerably complex topics, please consult the respective primary sources [DOCSIS MULPIv3.0] and [PKT-DQOS1.5]. The next section provides a summary overview of the PacketCable Multimedia architecture including each of the primary network elements and associated interfaces as further preparation for the technical protocol specification that follows.

5.2   Architecture

The PacketCable Multimedia technical report [MM-ARCH] describes an architecture framework and reference model for PacketCable Multimedia. This specification applies the model contained in the architectural framework and adds normative requirements to provide a scalable, interoperable solution suitable for deploying PacketCable Multimedia services.

5.2.1   Client Types

The PacketCable Multimedia tech report defines three kinds of client types:

- Client Type 1, which represents existing "legacy" endpoints (e.g., PC applications, gaming consoles) that lack specific QoS awareness or signaling capabilities. This client has no awareness of DOCSIS, CableHome, or PacketCable messaging, and hence no related requirements can be placed upon it. Client Type 1 communicates with an Application Manager to request service, and does not (cannot) request QoS resources directly from the MSO access network.

- Client Type 2 is similar to a PacketCable 1.x telephony MTA in that it supports QoS signaling based on the PacketCable DQoS specification [PKT-DQOS1.5].

- Client Type 3 directly requests QoS treatment from the access network without Application Manager interaction. This client is aware of IETF standards-based RSVP and uses this protocol to request access network QoS resources directly from the CMTS.

In the current release of this specification support is limited to Client Type 1. Consequently, this release of this specification supports only Scenario 1, the "Proxied QoS with Policy Push" scenario, described in [MM-ARCH]. Under this scenario, the Application Manager is responsible for requesting QoS resources on behalf of the client, and a Policy Server pushes the request down to the CMTS, which is the device actually responsible for setting up and managing the DOCSIS service flows required by the application.

5.2.2 PacketCable Multimedia Devices

In addition to the client (which typically resides at a subscriber's premises), PacketCable Multimedia requires several network elements residing in, or accessible to and trusted by, the MSO's network. In the process of describing these network elements throughout this specification we borrow heavily from standard IETF terminology and concepts. For a more thorough treatment of the overall PacketCable Multimedia architecture, including a discussion of underlying requirements and objectives, please refer to the accompanying technical report [MM-ARCH].

Since COPS [IETF RFC 2748] and COPS-PR [IETF RFC 3084] each use the terms Policy Enforcement Point (PEP) and Policy Decision Point (PDP) in substantially different interaction scenarios and since PacketCable Multimedia adds further nuances to these concepts (particularly in the definition of the Policy Server), it is occasionally confusing to think solely in terms of PEPs and PDPs to understand the responsibilities of the various components of the PacketCable Multimedia architecture. To attempt to alleviate this confusion, portions of this specification employ the notion of a Service Control Domain and a Resource Control Domain to draw distinctions in the type of policy that is being defined and enforced.

The Resource Control Domain (RCD) may be defined as a logical grouping of elements that provide connectivity and network resource level policy management along the packet forwarding paths to and from an end host. The RCD consists of CMTS and Policy Server entities whose responsibilities include management of resources along the packet forwarding paths.

The Service Control Domain (SCD) is defined as a logical grouping of elements that offer applications and content to service subscribers. The Application Manager resides in the SCD. Note that there may be one or more SCDs related to a single RCD. Conversely, each RCD may interact with one or more SCDs.
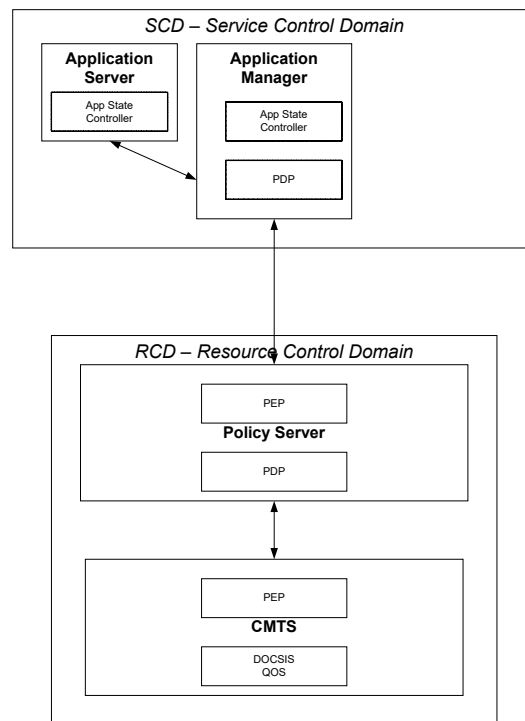
Figure 1 - Session & Resource Control Domains

In the PacketCable Multimedia architecture the fundamental role of the Application Server is to process requests on behalf of clients for network services and to convert them into session requests. The session request is then forwarded to the Application Manager for additional Service Control Domain processing.

In the PacketCable Multimedia architecture the fundamental role of the Application Manager is to maintain an application's session-level state and to enforce any SCD policies against client-driven session requests either directly

from the client or via the Application Server. If the client's session requests pass the Application Manager's SCD policy checks, the Application Manager converts the session request into a resource request and passes it on to the Policy Server for Resource Control Domain (RCD) policy checks. If the resource request fails the RCD policy check the Policy Server denies the resource request and the Application Manager consequently denies the client's session request. If, however, the resource request passes the Policy Server's RCD checks, the Policy Server forwards the request on to the CMTS for network-level admission control.

Fundamentally, the roles of the various PacketCable Multimedia components are:

- The Application Server is responsible for relaying client requests to the Application Manager.

- The Application Manager is responsible for application or session-level state and for applying SCD policy.

- The Policy Server is responsible for applying RCD policy and for managing relationships between Application Managers and CMTSs.

- The CMTS is responsible for performing admission control and managing network resources through DOCSIS Service Flows.

It may be useful here to clarify our use of the terms "admission control" and "policy authorization." For the purposes of this specification admission control is generally understood as the process of managing a finite pool of network-level resources (e.g., access network bandwidth, scheduler DOCSIS minislots, or CMTS resources supporting Gates and timers, etc.) and admitting requests against this pool. For performance reasons admission control is usually performed directly on network elements managing the packet forwarding path (such as the CMTS), though some sophisticated Policy Server implementations may choose to maintain state associated with network resources, thus supplementing and contributing to the admission control process.

In contrast, policy authorization is used here to describe higher-level aggregate usage policies (e.g., number of concurrent authorizations for a particular subscriber or service) which constitute an MSO's network management strategy. Policy authorization is almost always defined and enforced at the Policy Server.

The rest of this section describes each of these architectural components and their associated interfaces in more detail.

5.2.2.1   Application Server (AS)

The Application Server is a network entity that interfaces with the Application Manager that requests PacketCable Multimedia services on behalf of clients. The AS may reside on the MSO's network or it may reside outside of this domain and interact with the MSO network via a particular trust relationship. Similarly, the AS may be under direct control of the operator or it may be controlled by a third-party. Any given AS may communicate with one or more Application Managers.

The AS will communicate with a client via a signaling protocol that is outside the scope of this specification. Using this unspecified protocol, the AS authenticates and authorizes client requests based upon Service Control Domain policies. For client requests that pass these checks, the AS determines the particular QoS parameters necessary to deliver the service to the client, based upon its knowledge of the requested service. It then sends a request for these resources to the appropriate Application Manager, which may deny the request based upon additional Service Control Domain policies or may pass the request on to the Policy Server.

5.2.2.2   Application Manager (AM)

As noted in the preceding summary, the Application Manager is a network entity that defines SCD policies, coordinates subscriber-initiated requests for application sessions with access to the resources needed to meet those requests, and maintains application-level state.

The AM may reside on the MSO's network or it may reside outside this domain and interact with the MSO network via a particular trust relationship (typically defined by and enforced on the basis of a Service Level Agreement). Similarly, the AM may be under the direct control of the operator or it may be controlled by a third party. Any given Application Manager may communicate with one or more Policy Servers on the operator's network; likewise, one or more Application Managers may communicate with any given Policy Server on the operator's network (so long as an appropriate trust relationship exists).

The Application Manager may communicate with the Application Server using a signaling protocol as defined in 0. For Application Server requests the AM authenticates and authorizes the Application Server requests based upon Service Control Domain policies.

The Application Manager may communicate with a client via a signaling protocol that is beyond the scope of this specification. Using this unspecified protocol, the AM authenticates and authorizes client requests based on Service Control Domain policies.

For client and Application Server requests that pass these checks, the AM determines the particular QoS parameters necessary to deliver the service to the client, based on its knowledge of the requested service. It then sends a request for these resources to the appropriate Policy Server, which may deny the request based on network or RCD policy or may pass the request on to the CMTS for admission control and enforcement.

5.2.2.3   Policy Server (PS)

As discussed in [IETF RFC 2753], the policy management framework underlying PacketCable Multimedia is based on the work of the IETF's Resource Allocation Protocol (RAP) working group. Since the Policy Server is situated between the Application Manager and the CMTS, it simultaneously plays a dual role as a "proxy" for AM-initiated session requests and as a "sentry" for defining and enforcing Resource Control Domain policy.

As described in [IETF RFC 2753] and in keeping with the PacketCable 1.x DQoS model, the Policy Server serves as Policy Decision Point (PDP) in relation to the CMTS in that the Policy Server implements MSO-defined authorization and resource-management procedures. Conversely, the Policy Server assumes the role of Policy Enforcement Point (PEP) in relation to the Application Manager as it proxies Gate Control messages to and from the CMTS element.

To revisit the interaction scenario, the Application Manager issues policy requests to the Policy Server. The Policy Server acting as a "sentry" for these requests, and applies a set of policy rules that have been pre-provisioned by the MSO. Upon passing the checks, the Policy Server then acts as a "proxy" with respect to the Application Manager and the CMTS, forwarding the policy request and returning any associated response. Each policy request transaction must be processed individually.

Policy decisions may be based on a number of factors, such as:

- Parameters associated with the request and the status of available resources

- Identity of the particular client and associated profile information

- Application parameters

- Security considerations

- Time-of-day

The primary functions of the Policy Server include:

- A policy decision request mechanism, invoked by Application Managers

- A policy decision request 'policing' mechanism, enforcing installed Policy Rules

- A policy decision delivery mechanism, used to install policy decisions on the CMTS

- A mechanism to allow for the proxying of QoS management messages to the CMTS on behalf of the Application Manager

- An event recording interface to a Record Keeping Server that is used to log policy requests, which may in turn be correlated with network resource usage records

Since the Policy Server functions as a proxy between the AM and CMTS elements (with complementary client and server interfaces) some MSOs may elect to deploy multiple layers of Policy Servers and to delegate certain policy decisions among these servers in order to satisfy requirements associated with scalability and fault-tolerance.

5.2.2.3.1   Stateful & Stateless Policy Servers

There are two basic classes of Policy Servers—Stateful and Stateless. A Stateless Policy Server is a slight misnomer since it does maintain enough state to map Application Manager requests to the proper CMTS and maintain COPS session state, while a pure Stateless Policy Server maintains no state on any of the media sessions. Stateful Policy Servers come in several varieties—some participate in admission control and thus monitor the QoS attributes of active media sessions, some leave QoS and admission control to the CMTS but monitor time-based or volume-based service requests from the Application Manager, and some Policy Servers are somewhere between these extremes.

The reason there is a variety of Policy Server types is that there is a variety of environments that operators are trying to support. For example, some operators may wish to support PacketCable Multimedia over the same CMTSs that they use for PacketCable telephony, and they may want a single CMS/Policy Server that has a more global view of the network resources being used. On the other hand, some operators may wish to run a PacketCable Multimedia-only environment, or they may utilize simpler CMTS-driven mechanisms for partitioning PacketCable Multimedia and telephony resources. These simpler configurations have more modest requirements on the amount of state that a Policy Server maintains.

Policy Server state requirements can also be driven by the level of trust between the Policy Server and Application Manager; a Stateful Policy Server can more readily police Application Manager session control behavior than can a Stateless Policy Server. So a Stateful Policy Server may be more appropriate for operators supporting third party Application Managers. Other operators may rely on economics to enforce their trust relationships with Application Managers, or they may control the Application Managers themselves. In such cases a Stateless Policy Server may be more appropriate.

Since it is impossible to categorize all the various components of media session and network QoS state that a Policy Server is maintaining, the protocol is designed to be independent of this complexity. A Stateful Policy Server gleans PacketCable Multimedia media session information from the Application Manager requests it proxies; any other information it requires is gathered via mechanisms that are outside the scope of this specification. The CMTS and the Application Manager make no distinction as to the type of Policy Server to which they are connected, and the protocol is designed in such a manner that the type of Policy Server is transparent to the end point. The type of Policy Server is only of importance to the operator.

Since some types of Policy Servers attempt to assist with admission control and may have a larger view of the network and its resources, additional state synchronization issues may arise in design in a network which contains more than one of these types of Policy Servers. It is the responsibility of the operator to ensure that the efforts of these Policy Servers are not undermined by a network that includes other autonomous Policy Servers.

5.2.2.3.2   Modification of Requests and Responses by Policy Servers

Although nominally a part of the Resource Control Domain, the Policy Server can be an intermediary between the Service and the Resource Control Domains, in addition to its normal role of implementing MSO-defined authorization and resource management procedures. In either of these capacities it may modify the incoming request before forwarding it to the CMTS.

In acting as an intermediary between the SCD and RCD, the Policy Server may translate fields from formats or scales used in the SCD into formats or scales used in the RCD. For example, the Policy Server may modify the "priority" of a request coming from an Application Manager (especially important to do for an AM outside of the MSO network) so that this priority field uses a consistent scale throughout the operator's RCD. In its capacity as an intermediary, the Policy Server may use bidirectional translation—in other words, it should translate requests from the AM to the CMTS and "untranslate" the responses from the CMTS to the AM. This capability can be supported by stateful policy servers by remembering the original request, and it can be supported by stateless Policy Servers if the translation function is invertible.

Modification of certain objects, specifically the Classifier and Traffic Profile objects, may cause operational problems in the originating AM. As such, these objects MUST NOT be modified by the policy server. Aside from these exceptions, all other objects may be policed and modified at the PS's discretion based on provisioned policy rules.

5.2.2.4   Cable Modem Termination System (CMTS)

In describing the role of the CMTS network element, it is important to consider the relation among DOCSIS, PacketCable 1.x and PacketCable Multimedia functionality. While each of these suites of specifications addresses a specific set of functional requirements, each has also been defined in such a way that corresponding implementations may be constructed in a modular manner; either PacketCable 1.x or PacketCable Multimedia Gate Control may be layered on top of a DOCSIS 1.1 or greater CMTS foundation, with the option of adding additional, complementary functionality as business indicates. Further, it should be emphasized that it is a significant asset of the PacketCable architecture that both telephony and Multimedia variants employ considerable architectural similarity, leading to potential reuse in the underlying Gate management models.

The PacketCable Multimedia CMTS is a generalized version of the PacketCable 1.x CMTS that has been defined in order to deliver telephony services in PacketCable 1.x networks. The CMTS is responsible for fulfilling requests for QoS that are received from one or more Policy Servers. It performs this function by installing Gates, which are similar to the Gates defined in [PKT-DQOS1.5]; Gates allow the subscriber's cable modem to request network resources from the CMTS through the creation of dynamic DOCSIS flows with guaranteed levels of QoS. The CMTS also sends Event Messages detailing actual usage of QoS resources to the Record Keeping Server.

5.2.2.5   Record Keeping Server (RKS)

The PacketCable Multimedia Record Keeping Server fulfills a similar role to the RKS in PacketCable 1.x [PKT-EM1.5]. It receives Event Messages pertaining to policy decisions from the Policy Server and Event Messages pertaining to QoS resource usage from the CMTS.

In the PacketCable Multimedia architecture, the Record Keeping Server does not receive messages directly from the Application Manager. However, the Application Manager can embed opaque data in messages that it sends to the Policy Server, and these data can then be included in Event Messages that are subsequently sent to the RKS.

5.2.3   PacketCable Multimedia Interfaces

PacketCable Multimedia builds on the PacketCable 1.x suite of specifications. Where a PacketCable Multimedia interface has a corollary in PacketCable 1.x, PacketCable Multimedia uses the same protocol, or an extension of the same protocol.
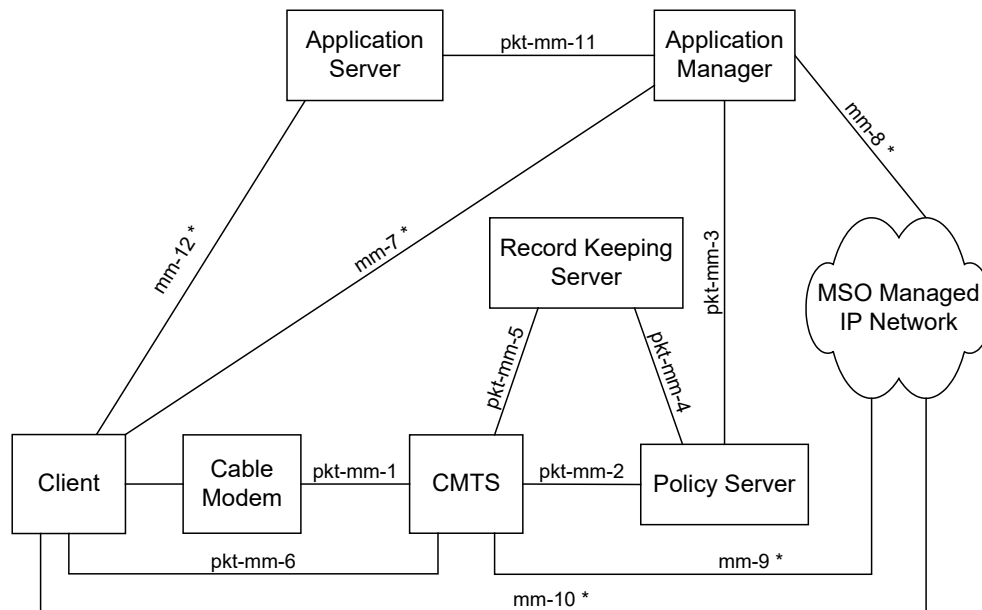
* Out of Scope

Figure 2 - PacketCable Multimedia Architectural Framework

Table 1 - PacketCable Multimedia Interfaces

Interface	Description	Notes
pkt-mm-1	CMTS – CM	The Cable Modem (CM) may request QoS from the CMTS via DOCSIS DSx signaling. Alternatively, the CMTS may instruct the CM to setup, teardown or change a DOCSIS service flow in order to satisfy a QoS request, again via DSx signaling.
pkt-mm-2	PS – CMTS	This interface is fundamental to the policy-management framework. It controls policy decisions, which may be: (a) pushed by the Policy Server (PS) onto the CMTS, or (b) pulled from the PS by the CMTS. The interface also allows for proxied QoS requests on behalf of a client. In some scenarios, this interface may also be used to inform the PS when QoS resources have become inactive.
pkt-mm-3	AM – PS	The Application Manager (AM) may request that the PS install a policy decision on the CMTS on behalf of the client. This interface may also be used to inform the AM of changes in the status of QoS resources.
pkt-mm-4	PS – RKS	The PS sends event messages to the Record Keeping Server (RKS) to track policy decisions related to QoS.
pkt-mm-5	CMTS – RKS	The CMTS sends the RKS event messages to track requests for and usage of QoS (e.g., service flow additions, changes, deletions, and volume metrics).
pkt-mm-6	Client – CMTS	The client may use this interface to directly request and manage QoS network resources. If authorized, these resources are provided by the CMTS.
mm-7	Client – AM	This interface may be used by the client to interact with the AM and to request and manage QoS resources indirectly. This interface is out of scope for this version of this specification.
mm-8	AM – Peer	The AM may use this interface to interact with some other entity that is part of the application in question. This interface is out of scope for this version of this specification.
mm-9	CMTS – MSO-Managed IP Network	This interface on the CMTS may be used in support of end-to-end QoS requests beyond the access network. This interface is out of scope for this version of this specification.
mm-10	Client – Peer	The Client may use this interface to interact with some other entity that is part of the application in question. This interface is out of scope for this version of this specification.
pkt-mm-11	AS – AM	The Application Server uses this interface to send network resource requests on behalf of the client to the AM. This interface may also be used to notify the AS of changes in the status of the network resources.
mm-12	Client – AS	This interface may be used by the client to interact with the AS and to request applications, which indirectly requests network resources. This interface is out of scope for the current effort.

5.2.3.1   Client and Application Manager Interface (mm-7)

The interface between the client and the Application Manager is out of scope. Typically, the Application Manager will, through some means beyond the scope of this specification, authenticate the client and ensure that the client is entitled to the Multimedia service. For example, the client may login to a web page and request service by providing a username and password. However it is accomplished, the Application Manager must be able to identify unambiguously the cable modem(s) to which service is to be delivered, since this information must be made available to the network operator before QoS can be delivered.

5.2.3.2   Application Manager and Policy Server Interface (pkt-mm-3)

This interface corresponds to the PacketCable 1.x interface between a Call Agent and a Gate Controller. In PacketCable 1.x this is a hidden, non-testable interface, and so there are no pre-existing protocol requirements on this interface.

PacketCable Multimedia requires the use of COPS [IETF RFC 2748] on this interface. In order to simplify the architecture and to allow for multiple levels of Policy Server elements between the Application Manager and CMTS, this interface mirrors as far as possible the interface between the Policy Server and the CMTS. Although the

Application Manager is the one requesting resource authorization from the Policy Server, it actually issues this request in a COPS Decision message, instead of a COPS Request message. This allows the interface between the Application Manager and the Policy Server to appear identical to the interface between the Policy Server and the CMTS. The Application Manager is the PDP relative to the Policy Server, and the Policy Server is the PEP relative to the Application Manager.

When an Application Manager agrees to provide service to a client, it sends a COPS Decision that contains (at least) the following information in the form of COPS objects:

- Identity of the Application Manager making the request

- Identity of the client(s) to whom service is to be provided

- RSVP FlowSpec(s) specifying traffic envelope(s) for the session

In the reply from the Policy Server, the server includes an authorization token, the Gate-ID, which is provided to it by the CMTS.

5.2.3.3   Policy Server and CMTS Interface (pkt-mm-2)

This interface is essentially identical to the equivalent interface (between CMTS and Gate Controller) in PacketCable 1.x. As in PacketCable 1.x, COPS is used to transfer policy information between the Policy Server and the CMTS. The CMTS acts as a COPS PEP and the Policy Server acts as a COPS PDP. Following the PacketCable 1.x model, the Policy Server initiates communication for a Multimedia session by sending a DQoS Gate-Set message (which is an unsolicited COPS DECISION message) to the CMTS.

This message contains (at least):

- Application Manager ID

- Subscriber ID

- GateSpec

- FlowSpec(s)

- Classifier

The CMTS responds, as in DQoS, with either Gate-Set-Ack or Gate-Set-Err (both of which are COPS REPORT messages).

If the CMTS responds positively (i.e., with a Gate-Set-Ack), it includes a GateID. As in PacketCable 1.x, the GateID acts as an authorization token. Unlike PacketCable 1.x, the token is not ultimately passed to the client (since Client Type 1 endpoints have no knowledge of PacketCable); instead it is held by the Policy Server (if stateful) and by the Application Manager, thus allowing them to issue commands pertaining to this session to the CMTS, either directly in the case of the Policy Server, or indirectly via the Policy Server in the case of the Application Manager.

5.2.3.4   Record Keeping Server and Policy Server Interface (pkt-mm-4) and
Record Keeping Server and CMTS Interface (pkt-mm-5)

The interfaces into the Record Keeping Server from the Policy Server and the CMTS are identical to the equivalent interfaces (from CMS and CMTS, respectively) in PacketCable 1.x ([PKT-EM1.5]). These interfaces are used to carry PacketCable Event Messages, which use RADIUS formatting. In PacketCable Multimedia, Event Messages carry detailed information pertaining to the service delivered, including the exact time that service flows are created and deleted and (optionally) the amount of traffic that passed over the service flow while it was in existence.

5.2.3.5   Client and Application Server Interface (mm-12)

The interface between the client and the Application Server is out of scope. Typically, the Application Server will, through some means beyond the scope of this specification, authenticate the client and ensure that the client is entitled to the Multimedia service. However it is accomplished, the Application Server must be able to identify the subscriber to which service is to be delivered, since this information must be made available to the network operator before QoS can be delivered.

5.2.3.6   *Application Server and Application Manager Interface (pkt-mm-11)*

The interface between the Application Server and the Application Manager is defined in [BPI+]. PacketCable Multimedia requires the use Web Services on this interface as specified in [BPI+]. Application Servers use this interface to send network resource requests on behalf of clients to the AM. The interface may also be used by the AM to send notifications to the AS of changes in the status of network resources.

When an Application Server agrees to forward a service request for a client, it sends a Web Service message that contains (at least) the following information in the form of message arguments:

> Subscriber Identifier
> Service Name

The interface supports a number of optional arguments that may be used to provide more granularity when requesting the service.

5.2.4   State Information

In this section, we provide an overview of state location in a PacketCable Multimedia system. In addition to maintaining detailed state information, devices send information about state transitions to the Record Keeping Server for purposes such as billing, fraud detection, session reconstruction, etc.

5.2.4.1   *Application State*

The Application Manager is at all times responsible for maintaining detailed knowledge of the state of the application media session. How it does so in detail is beyond the scope of this specification, but it is important to recognize that no devices other than the Application Manager are required or expected to maintain any knowledge of the application state.

The Application Manager may, however, report session state by proxying such information via the Policy Server to the Record Keeping Server. In addition, some coarse state information (such as the fact that resources have been requested) is automatically sent from the Policy Server to the Record Keeping Server.

5.2.4.2   *QoS Resource State*

The CMTS naturally is aware of the detailed state of flows that it manages. The Policy Server (if it is a stateful Policy Server) may also maintain a notion of the state of QoS resources on a single CMTS; it may also collate the state information over several CMTSs so that it (and only it) knows the QoS state of the entire system. This may be important, for example, if an operator has instituted a policy whereby a particular application is not to be permitted to consume more than a specified percentage of the total system resources. In a network with only stateless Policy Servers, the CMTSs are the only devices to maintain QoS state information. Since stateless Policy Servers do not maintain GateIDs, they cannot even interrogate a CMTS to obtain information about a particular Multimedia session.

Whenever a QoS Resource transitions from one state to another, and whenever a QoS Resource is deleted, a corresponding Event Message is sent from the CMTS to the Record Keeping Server.

5.2.5   Persistent Gates

There are cases where a service that is implemented for a subscriber requires that the QoS resources allocated for the that service be available after a CM has rebooted. An example of these services might be service restrictions/downgrades for Acceptable Use Policy Management (AUPM), or service upgrades for higher service tier trials, both of which use PCMM to dynamically modify the subscriber's service without having to change the cable modem configuration file. To enable this behavior, a new feature called Persistent Gates has been defined in this specification. The persistent gate feature allows an Application Manager to request that a Committed Gate be re-created should the cable modem be rebooted or go offline for a period of time. Traditional, non-persistent, Gates are normally removed if a cable modem reboots or goes offline. For a persistent gate, if a cable modem reboots, then any gates identified as persistent are re-created when the modem comes back on-line. This minimizes the time for the subscriber's service to be in its default state (as per CM configuration file) and helps restore services as quickly as possible. For example, persistent gates can be used to reenable any AUPM service restrictions or service

modifications to allow for a premium service flow, without any additional signaling from the Policy server. The method of signaling that a Gate should be persistent is defined in Section 6.4.2.5.

If the CMTS does not support this Persistent Gate feature or resources are not available, the appropriate Gate State Errors should be returned during the gate creation procedure (see 6.4.2.14) When the persistent gate feature is not supported by a CMTS, an application is forced to periodically check to see if gates that should be maintained across a cable modem reboot still exist and if a gate has been removed, detect when the modem comes back on-line and re-create the gate. This results in additional messaging to CMTSs and back-office systems and increases the window when a subscriber is not receiving the desired service handling.

When a cable modem goes offline that has associated gates in a committed state identified as being persistent, the CMTS MUST save all the gate information and set their state to Idle (6.4.2.15), so that they can be recreated when the modem comes back online. The CMTS MAY delete a gate in the persisted state upon receiving a Gate-Delete command. Gates that are in a authorized or reserved state, but not yet committed MUST NOT be treated as persistent by the CMTS even if those Gates are identified as persistent.

When a cable modem that had persistent gates comes back online, the CMTS attempts to recreate the persistent gate with the same stored GateId by performing the admission control steps (see 6.4.2.5). If resources are not available or the gate recreation fails, the CMTS MUST generate a Gate-State-Report message to inform the Application Manager that the persistent gate could not be readmitted with an error code (38) Persistent Gate resources unavailable (see 6.4.2.15).

The CMTS MUST delete persistent gates during CMTS reboot. The CMTS MAY delete persisted Gate information for offline modems after a configurable time period (see [CCAP-OSSIv3.1]).

6 AUTHORIZATION INTERFACE DESCRIPTION

This section describes the interface between the Application Manager and the Policy Server, and the interface between the Policy Server(s) and the CMTS.

The interface between the Application Manager and the Policy Server is translationally symmetric to the interface between the Policy Server and the CMTS. The interfaces are used to pass authorization, reservation and activation information to the CMTS, to provide state information from the CMTS to the Policy Server, and from the Policy Server to the Application Manager.

The Application Manager is the PDP for the Service Control Domain. The Policy Server is the PEP with respect to the Application Manager and applies Resource Control Domain policies. The Policy Server is a PDP with respect to the CMTS, and the CMTS is the PEP with respect to the Policy Server and lies in the actual packet forwarding path.

This section describes the use of the COPS protocol to transport PacketCable QoS messages between the Application Manager and Policy Server, and between the Policy Server and CMTS.

6.1 Gates: The Framework for QoS Control

A PacketCable Multimedia Gate is a logical representation of a policy decision that has been installed on the CMTS. A Gate is used to control access by a single IP flow to enhanced QoS Services provided by a DOCSIS cable network. Gates are unidirectional; a single Gate controls access to a flow in either the upstream or the downstream direction, but not both. For a bi-directional IP session, two Gates are required, one for upstream and one for downstream, each identified by a unique GateID. It is important to recognize that this is a fundamental difference from PacketCable 1.x, in which a single GateID may reference both an upstream and a downstream Gate.

In PacketCable Multimedia, each Gate has a separate GateID. The Gate defines the authorization, reservation and committed envelopes to be used by the CMTS to perform authorization, reservation and commit operations.

Gates can be further qualified as either unicast or multicast. Fundamentally the two Gates are as described above with one key differentiator between them—the destination IP address indicated in the classifier object. In a Unicast Gate the destination IP address (v4 or v6) contained in the classifier identifies a unicast IP address (or a range of unicast IP addresses). A Multicast Gate contains a destination IP address (v4 or v6) in the classifier identifying a multicast IP address. See [IETF RFC 4291] for IPv6 unicast and multicast IP address definitions and [IETF RFC 4291] for IPv4 multicast address assignments.

Throughout this document, unless specifically called out as either a Unicast Gate or Multicast Gate, the reference to Gate applies to both Gate types.

In all Scenarios, the CMTS MUST perform admission control checks of the envelopes to make sure that the committed envelope is less than or equal to the reserved envelope, and the reserved envelope is less than or equal to the authorized envelope. (See [IETF RFC 3171] for specific DOCSIS admission control requirements.)

In the 'Proxied QoS Policy Push' (Scenario 1) model, the information in a Gate is used by the CMTS to create the DOCSIS Service Flow directly, after the CMTS performs the necessary admission control checks of the envelopes. In the other two models outlined in [MM-ARCH], 'Client requested QoS with Policy Push' (Scenario 2) and 'Client requested QoS Policy Pull' (Scenario 3), the CMTS uses the Gate information to perform admission control of the client requested resources; the CMTS does not initiate creation of the flows. The Application Manager is responsible for issuing Gate messages to the Policy Server and the Policy Server is responsible for applying policy rules, and then issuing Gate Control messages to the CMTS.

A Gate consists of the following elements, which are described later in this section:

- GateID
- AMID
- SubscriberID
- GateSpec
- Classifier
- Traffic Profile

- Event Generation Info (optional)
- Time-Based Usage Limit (optional)
- Volume-Based Usage Limit (optional)
- Opaque Data (optional)
- UserID (optional)
- SharedResourceID (optional)

GateID is the handle for the Gate. The GateID is assigned by the CMTS and is used by the Application Manager, Policy Server, and client to reference the Gate.

AMID is the handle that identifies the Application Manager and Application Type.

SubscriberID uniquely identifies the Client for which the policy is being set.

GateSpec describes specific authorization parameters defining a Gate (i.e., QoS limits, timers, etc.).

Classifier describes the IP flow(s) that will be mapped to the DOCSIS Service Flow.

Traffic Profile describes the QoS attributes of the Service Flow used to support the IP flow.

Event Generation Information contains information used by the CMTS for the purpose of accounting and usage reporting.

Volume-Based Usage Limit defines a volume cap for traffic traversing the flow associated with the Gate.

Time-Based Usage Limit describes a time cap limiting the duration of the flow associated with the Gate.

Opaque Data represents a general-purpose object that remains opaque to the CMTS and PS elements, but which may contain data that is significant to the AM. This optional object, if provided by the AM, is retained at the CMTS and returned in all associated responses (see Section 6.4.2.11).

UserID identifies the User for which the policy is being set.

SharedResourceID indicates the use of a shared underlying resource in fulfilling the request, and uniquely identifies that resource. The SharedResourceID could be used by the Application Manager and Policy Server to aid in understanding true resource allocation. A single SharedResourceID may be associated with multiple gates, allowing an accurate summary of resource allocations while preserving the gate's role in identifying a specific policy decision.

These elements are communicated to the Policy Server and the CMTS via COPS objects and are described in greater detail later in this section. During the installation of the Gate, the information above is communicated to the CMTS. After the installation is complete, a DOCSIS Service Flow can be created. After the creation of the DOCSIS Service Flow, the Gate has associated with it an additional element, the DOCSIS Service Flow.

A Gate transitions through multiple states. In Scenarios 2 and 3, where the client entity is responsible for reserving and then activating the DOCSIS Service Flows, a Multimedia Gate behaves in a manner very similar to a PacketCable 1.x DQoS Gate. When the Policy Server installs the Gate onto the CMTS, the Gate is said to be in an 'Authorized' state. It remains in this state until explicitly deleted by the Policy Server (or, less likely, it is deleted for some reason by the CMTS itself) or until a dynamic flow request from the client arrives.

When the client requests that a dynamic Service Flow be added, it presents the GateID as an authorization token. The CMTS uses the GateID to perform admission control on the DOCSIS dynamic flow against the authorized envelope defined by the Gate. In Scenario 1, the Policy Server instructs the CMTS to transition between the states on behalf of the Application Manager, and the CMTS is the entity responsible for initiating and tearing down DOCSIS Service Flows. The State Transition section in this document describes this behavior. When the CMTS is instructed to tear down a DOCSIS Service Flow, its associated Gate remains until explicitly deleted by the PS/AM or until it times out and its resources are reclaimed by the CMTS (see Section 6.5.8). In contrast, however, when the PS/AM deletes a Gate, the CMTS will delete the associated DOCSIS Service Flow.

6.1.1 Gate Identification (GateID)

A GateID is an identifier that is locally allocated by the CMTS where the Gate resides. A GateID MUST be associated with only one Gate. Whereas the PacketCable 1.x DQoS Gate Control model generally assumed a pair of

unidirectional Gates (one upstream and one downstream) per GateID in support of a typical two-way voice session, here the Gate/GateID relationship is explicitly one-to-one, so that it is easier to support a wide range of Multimedia services.

When the Application Manager issues a Gate-Set request, this triggers the Policy Server to issue a Gate-Set message to the CMTS. When the CMTS responds with an acknowledgment containing the GateID, the Policy Server forwards this response including the GateID back to the Application Manager. Note that since there can be a many-to-many relationship between a PS and CMTS, the GateID assigned by one CMTS cannot be guaranteed to be unique across the network, so the PSs may use the AMID of the AM along with the SubscriberID and GateID in order to uniquely identify the Gate.

An algorithm that may be used to assign values of GateIDs is as follows. Partition the 32-bit word into two parts, an index part, and a random part. The index part identifies the Gate by indexing into a small table, while the random part provides some level of obscurity to the value. Regardless of the algorithm chosen, the CMTS SHOULD attempt to minimize the possibility of GateID ambiguities by ensuring that no GateID gets reused within three minutes of its prior closure or deletion. For the algorithm suggested this could be accomplished by simply incrementing the index part for each consecutively assigned GateID, wrapping around to zero when the maximum integer value of the index part is reached.

6.1.2   Application Manager Identification (AMID)

The AMID consists of two fields: the Application Manager Tag and Application Type. Each Application Manager is pre-provisioned with an Application Manager Tag that is unique within the universe of a single service provider. The Application Manager may also be pre-provisioned with a set of Application Type values that can be used to identify the particular application that a gate is associated with. The Application Manager includes the AMID in all messages that it issues to the Policy Server. The Policy Server transparently passes this information to the CMTS via Gate Control messages. The CMTS MUST return the AMID associated with the Gate to the Policy Server. The Policy Server uses this information to associate Gate messages with a particular Application Manager and Application Type.

The Application Manager Tag MUST be a globally unique value assigned to the Application Manager by the service provider. The Application Manager MUST use the assigned Application Manager Tag in all its interactions with Policy Servers. Note that since the Application Manager may be operated by a third party, and a single Application Manager could interact with multiple service provider operators, a single physical Application Manager may be provisioned with multiple Application Manager Tags, and multiple Application Type sets (one for each configured Application Manager Tag).

6.1.3   Subscriber Identification (SubscriberID)

The SubscriberID, consisting of the IPv4 or IPv6 address of either the CM or client CPE device (either directly connected to the CM or on a routable network behind the CM), identifies the user requesting the service. In complex network environments this address may be used to route Gate Control messages between a number of Policy Servers and to determine which CMTS is providing service to a particular endpoint. In addition to the IPv4 or IPv6 address, a subscriber may also be identified via a FQDN or some opaque data (object defined below) relevant to the service in question.

For a Multicast Gates the CMTS uses the SubscriberID to decide where the Multicast replication needs to be created. The CMTS treats the SubscriberID as the source IP address of a JoinMulticastSession [DOCSIS MULPIv3.0]. If the SubscriberID is on an IP subnet that is not directly connected to the CMTS, the CMTS MAY reject the Gate as having an invalid SubscriberID see 6.4.2.14 PacketCable Error.

6.1.4   Gate Specification (GateSpec)

The GateSpec describes some high-level attributes of the Gate, and contains information regarding the treatment of other objects specified in the Gate message. Information contained in a GateSpec is outlined below:

- SessionClassID
- Direction
- Authorized Timer

- Reserved Timer
- Committed Timer
- Committed Recovery Timer
- DSCP/TOS Overwrite
- DSCP/TOS Mask

SessionClassID provides a way for the Application Manager and the Policy Server to group Gates into different classes with different authorization characteristics. For example, one could use the SessionClassID to represent some prioritization or preemption scheme that would allow either the Policy Server or the CMTS to preempt a pre-authorized Gate in favor of allowing a new Gate with a higher priority to be authorized.

Direction indicates whether the Gate is for an upstream or downstream flow. Depending on this direction, the CMTS MUST reserve and activate the DOCSIS flows accordingly. For Multicast Gates the CMTS needs to only support flows or gates in the downstream direction.

Authorized Timer limits the amount of time the authorization must remain valid before it is reserved (see Section 6.2).

Reserved Timer limits the amount of time the reservation must remain valid before the resources are committed (see Section 6.2).

Committed Timer limits the amount of time a committed service flow may remain idle.

Committed Recovery Timer limits the amount of time that a committed service flow can remain without a subsequent refresh message from the PS/AM once the PS/AM has been notified of inactivity (see Section 6.2).

DSCP/TOS Overwrite field can be used to overwrite the DSCP/TOS field of packets associated with the DOCSIS Service Flow that corresponds to the Gate. This field may be unspecified in which case the DSCP/TOS field in the packet is not over-written by the CMTS. This field may be used in both the upstream and downstream directions. The CMTS MUST support DSCP/TOS overwrite for upstream gates. The CMTS MAY support DSCP/TOS overwrite for downstream gates. If DSCP/TOS is enabled in a downstream gate and the CMTS does not support that function, then the field is ignored. The manner in which the CMTS interprets the DSCP/TOS Overwrite & Mask fields and transforms it into the corresponding DOCSIS Service Flow Parameters is defined in Section 6.4.2.5.

6.1.5   Classifier

A classifier MUST be defined for a Gate. Additional classifiers may be included in the original Gate-Set. Classifiers may be added or deleted in a subsequent Gate-Set. Compliant implementations MUST be able to support a minimum of four classifiers when processing a Gate-Set message for Unicast Gates. Compliant implementations MUST support a minimum of one classifier when processing a Gate-Set message for Multicast Gates. The AM and PS MUST NOT mix unicast and multicast destination IP addresses for a single Gate. The classifier identifies the IP flow that will be mapped to the DOCSIS service flow associated with the Gate. In Scenario 1, when the CMTS creates the dynamic flow, it MUST use the Gate classifier as the classifier for the DOCSIS Service Flow according to the rules specified in Sections 9.3.4.1 and 9.4.3.1.

There are three types of classifiers defined in this specification; a Classifier, an Extended Classifier, and an IPv6 classifier, which are described below. For IPv4 traffic, an Application Manager SHOULD use the Extended Classifier to properly match the traffic it is interested in unless it receives a Gate-Set-Err with either Error-Code 6 (Missing Required Object) and Error-Subcode of 6 | 1 (S-Num | S-Type) or Error-Code 7 (Invalid Object) and Error-Subcode of 6 | 2. This would occur if the AM is interacting directly with a Policy Server or indirectly with a CMTS that does not support the Extended Classifier. In this case, the AM may resend the Gate using the Classifier object instead of the Extended Classifier object. The Classifier object is retained for legacy purposes. That being said, the legacy Classifier object will be deprecated from the specification in a future release.

For IPv6 traffic, an Application Manager MUST use an IPv6 Classifier.

For Multicast Gates (where the Destination IP Address in the classifier identifies a specific Multicast IP address) the AM encodes the classifier (any of the three classifier types) as follows:

- The Source IP address & the Mask/Prefix Length combination MUST resolve to a specific valid Unicast IP address value, i.e., not resolve into a range of addresses. If the Source IP address and the Mask/Prefix

Length combination resolve to the All-Zeros IP Address, it indicates that all Source IP addresses are permitted, i.e., Any Source Multicast (ASM).

- The Destination IP address and the Mask/Prefix Length combination MUST resolve to a single specific Multicast address.

- The Source and Destination Port Number fields in the classifier MUST be ignored by the CMTS for a Multicast Gate.

The CMTS MUST reject a Multicast gate request which does not meet the above stated Multicast gate classifier rules with error code 7 (invalid object) and the Error-subcode identifying the invalid classifier object.

A PS MUST support all three types of classifiers specified here. A CMTS MUST support both legacy and Extended Classifiers. A CMTS that supports DOCSIS 3.0 MUST support the IPv6 Classifier. For IPv4 classification, an Application Manager MUST use either the legacy or the Extended classifier type per Gate-Set, and MUST continue to use that type for all subsequent Gate-Sets for that Gate. If both IPv4 and IPv6 based classification are required in a Gate-Set, an Application Manager MUST use the Extended Classifier for the IPv4 classification and logically the IPv6 Classifier for the IPv6 classification.

Throughout this specification, unless stated otherwise, the "classifier" refers generically to either the Classifier object, Extended Classifier object, or IPv6 Classifier object, depending on which type was used in the Gate-Set.

A (legacy) Classifier is an eight-tuple:

- Protocol
- Source IP
- Source Port
- Destination IP
- Destination Port
- Priority
- DSCP/TOS Field
- DSCP/TOS Mask

An Extended Classifier is defined by the following tuple:

- Protocol
- IP Source Address
- IP Source Mask
- Source Port Start
- Source Port End
- IP Destination Address
- IP Destination Mask
- Destination Port Start
- Destination Port End
- Priority
- DSCP/TOS Field
- DSCP/TOS Mask
- ClassifierID
- Activation State
- Action

An IPv6 Classifier is defined by the following tuple:

- Tc-low
- Tc-high
- Tc-mask
- Flow Label
- Next Header Type
- Source Prefix Length

- Destination Prefix Length
- IPv6 Source Address
- IPv6 Destination Address
- Source Port Start
- Source Port End
- Destination Port Start
- Destination Port End
- ClassifierID
- Priority
- Activation State
- Action

Protocol field, in a legacy Classifier or Extended Classifier, identifies the type of protocol (e.g., IP, ICMP, etc.). The Next Header Type field serves a similar function in the IPv6 Classifier.

Source IP, IP Source Address, or IPv6 Source Address (in the case of Extended Classifier or IPv6 Classifier) is the IP address (as seen at the CMTS) of the originator of the IP flow, while Destination IP, IP Destination Address or IPv6 Destination Address is the termination point for the IP flow. When using an Extended Classifier, a subnet can be used to enable classifying multiple IP addresses with a single Extended Classifier object by the use of the IP Source Mask and/or IP Destination Mask fields. Similarly, when using the IPv6 Classifier, the Source Prefix Length and Destination Prefix Length indicate how many high order bits in the corresponding IPv6 Address to consider in determining a match.

Source Port and Destination Port specify the UDP or TCP ports for the IP flow. When using an Extended Classifier or IPv6 Classifier, the Source Port (Start/End) and Destination Port (Start/End) specify the UDP or TCP port ranges for matching the IP flow.

Priority may be used to distinguish between multiple classifiers that match a particular packet. This is typically set to a default value since classifiers are generally intended to be unique.

When using the legacy Classifier or Extended Classifier, DSCP/TOS field identifies the DSCP/TOS field that must be matched for packets to be classified onto the IP flow. To provide for maximum flexibility in defining a network management strategy, an accompanying mask is defined which determines which bits in the DSCP/TOS byte are to be used as filters in classifying packets. This allows for both DiffServ and TOS strategies (which each define and use separate bits within this byte).

When using IPv6 Classifier the Traffic Class Range and Mask fields, tc-low, tc-high, and tc-mask, allow matching on the IPv6 Traffic Class value. An IPv6 packet with IPv6 Traffic Class value "ip-tc" matches this parameter if tc-low ≤ (ip-tc AND tc-mask) ≤ tc-high. If the tc-mask field is set to 0, then comparison of the IPv6 packet Traffic Class byte for this entry is irrelevant.

Note: The value 0x3F for tc-mask will exclude the Explicit Congestion Notification in [IETF RFC 2474] bits from the comparison, and hence will result in classification based on DSCP [IETF RFC 2474].

The Flow Label field matches the Flow Label used in the IPv6 header. The 20 least significant bits represent the 20-bit IPv6 Flow Label while the 12 most significant bits are ignored. If this parameter is set to 0, then comparison of IPv6 flow label for this entry is irrelevant.

A classifier MAY have wild-carded fields (indicated by zeroed fields, unless otherwise specified), but care must be taken so that multiple IP flows do not unintentionally match the same classifier, which can lead to unexpected results.

Fields unique to the Extended Classifier and IPv6 Classifier:

- The ClassifierID field is used as an identifier for the classifier. It can be used to reference an existing classifier when making changes to that classifier.

- The Activation State field is used to instruct the CMTS to either activate or inactivate the classifier.

- The Action field is used to add, replace, or delete the classifier.

6.1.6   Traffic Profile

There are four basic ways to express the Traffic Profile for a Gate:

1. FlowSpec

2. DOCSIS Service Class Name, or

3. DOCSIS-Specific Parameterization

4. Upstream Drop

The Policy Server or Application Manager MUST define the Traffic Profile for a Gate using one of the following: (1) the FlowSpec, (2) DOCSIS Service Class Names, (3) DOCSIS-Specific Parameters, or (4) Upstream Drop. All envelopes used in a Traffic Profile MUST be the same type, i.e., either FlowSpec, DOCSIS Service Class Names, DOCSIS-Specific Parameters, or Upstream Drop.

There MUST be at least one set of Traffic Profile parameters specified when the Gate is first being installed. The Policy Server and Application Manager MAY specify a second set to represent the reserved envelope, and a third set to represent the committed envelope. If the CMTS is told to immediately create a dynamic flow upon receipt of a Gate-Set message (via the presence of the reserved or committed envelopes), the CMTS MUST use the reserved and committed envelope Traffic Profile parameters to perform the DOCSIS messaging to create the flow, in the direction specified by the Direction field in the GateSpec (provided the request is authorized and sufficient resources exist to satisfy the request). When told to transition into the Committed state, the CMTS MUST use the Traffic Profile to activate the DOCSIS Service Flow. As an optimization, the Policy Server MAY tell the CMTS to perform all three actions (authorize, reserve and commit) on behalf of the Application Manager via a single Gate Control message. Alternatively, the PS/AM MAY issue separate Gate-Set messages to tell the CMTS to authorize and reserve and then to commit via a subsequent Gate-Set message.

6.1.6.1   FlowSpec

The FlowSpec object contains RSVP FlowSpecs that are used to describe the Traffic Profile of the IP flow. The FlowSpec object can contain multiple RSVP FlowSpecs:

- FlowSpec that defines the authorization resource envelope against which future reservations can be made.

- FlowSpec that defines the reserved envelope against which future commit requests can be made.

- FlowSpec that defines the resources to be committed.

RSVP FlowSpecs support two types of services: controlled load [IETF RFC 2211] and guaranteed [IETF RFC 2212]. The main difference between the two types of services is discussed in Section 8. The two types of services are distinguished based on the FlowSpec Service number, which is specified in the RSVP FlowSpec. Service number 5 is for controlled load, and Service number 2 is for guaranteed. A controlled load service MUST contain only the TSpec token bucket parameters, and not the RSpec. A guaranteed service MUST contain both the TSpec and the RSpec.

Please refer to Section 8 for information on how to explicitly map RSVP parameters into DOCSIS parameters. When deriving the DOCSIS parameters using the RSVP FlowSpec parameters, there are some DOCSIS parameters that are highly approximated. If the approximations do not give the Policy Server or Application Manager the control it desires, the PS/AM MAY use the other methods of defining the Traffic Profile, which includes the ability to define some DOCSIS-specific parameters. These parameters allow the Policy Server or Application Manager to fine-tune the standard mapping of FlowSpecs to DOCSIS parameters.

6.1.6.2   DOCSIS Service Class Name

The DOCSIS Service Class Name indicates the DOCSIS Service Class to be used to describe the QoS attributes. A CMTS MUST support DOCSIS Service Class Names.

DOCSIS Service Class Name enables one to use pre-provisioned DOCSIS QoS parameters on the CMTS. On the CMTS, one can configure DOCSIS Named Service Classes with different DOCSIS QoS profiles, then reference the DOCSIS Service Class Name in the Gate to indirectly associate a QoS profile with a particular Gate. When using the DOCSIS Service Class Name traffic profile, values in the GateSpec, with the exception of the direction flag,

MUST overwrite the equivalent defined parameter value in the Service Class Name. For example, if T3 is 30 seconds in the GateSpec and the Service Class Name defined the Timeout for Active QoS Parameters to 20, the value in the GateSpec, i.e., 30, will be used. This currently applies to the GateSpec DSCP/TOS Overwrite, DSCP/TOS Mask, T2 and T3+T4, which map to the DOCSIS tos-or-mask, tos-and-mask, Timeout for Admitted QoS Parameters and Timeout for Active QoS Parameters respectively. If the CMTS receives a GateSpec with a direction flag that does not match the direction associated with the Service Class Name specified in the Gate-Set, the CMTS MUST send a Gate-Sate-Err with error code 11 (Undefined Service Class Name).

For more information on DOCSIS Service Classes please refer to section 7.5.3 of the DOCSIS specification [DOCSIS MULPIv3.0].

6.1.6.3   DOCSIS Specific Parameterization

The third way of defining the Traffic Profile consists of using DOCSIS-specific Traffic Profile; this allows the Application Manager to explicitly specify the DOCSIS parameters of the DOCSIS flow. If the Application Manager wishes to use this third way of defining a Traffic Profile, it MUST include an object containing the DOCSIS Specific Parameters.

All DOCSIS Service Flow Scheduling types are supported via several different Traffic Profile S-Types. Each S-Type has a different encoding of the DOCSIS-specific parameters relevant to that Service Flow Scheduling type. For further details regarding DOCSIS-specific parameterization refer to Section 6.4.2.7.

6.1.6.4   Upstream Drop

An Upstream Drop Traffic Profile represents a null service flow for upstream traffic filtering at the CM. DOCSIS 3.0 refers to a null service flow at the CM as an Upstream Drop Classifier (UDC). An AM can push an Upstream Drop Traffic Profile to the CMTS, via the PS, which pushes the UDC to the CM. When a CM installs a UDC all data packets meeting the criteria of the classifier are dropped at the CM.

6.1.7   Event Generation Info

This object contains information relevant to the CMTS in support of accounting and billing functions. Attributes include:

- Primary Address: Port of the Primary Record Keeping Server to which the CMTS MUST send event records.

- Secondary Address: Port of the Secondary Record Keeping Server, the CMTS MUST use as specified in [DOCSIS MULPIv3.0] if the primary is unavailable.

- BillingCorrelationID, which the CMTS MUST pass to the Record Keeping Server with each event record.

Omission of the Event Generation Info object indicates that the CMTS MUST NOT generate Event Messages for a specific Gate.

There are two versions of the Event Generation Info object: an IPv4 Event Generation Info object and an IPv6 Event Generation Info object to accommodate IPv4 and IPv6 Address values respectively.

6.1.8   Time-Based Usage Limit

This object specifies amount of time a Gate can remain committed before meeting the time limit threshold for this Gate. The CMTS is not responsible for enforcing time limits, but MUST store this object and return it upon request.

6.1.9   Volume-Based Usage Limit

The Application Manager uses the Volume-Based Usage Limit to signal the CMTS to generate a Gate Control message when the specified amount of data has traversed the Gate. The CMTS is not responsible for enforcing volume limits but MUST signal to the PS/AM when a volume limit is reached.

6.1.10 Opaque Data

Opaque Data consists of general information that a Policy Server or Application Manager can store on a CMTS. This data remains opaque to the CMTS, but contains information useful for the PS or AM. If provided by the PS/AM, the CMTS will return this object in all responses (see Section 6.4.2.11).

6.1.11 Gate Time Info

Gate Time Info contains a timestamp representing the time the Gate was committed. This may be queried and used by a Policy Server or Application Manager to enforce time-based network policy.

6.1.12 Gate Usage Info

Gate Usage Info consists of an octet counter indicating the number of bytes of data transmitted over this Gate (see Section 6.4.2.13). Analogous to the Gate Time Info object, this information may be used by a Policy Server or Application Manager to enforce volume-based network policy.

6.1.13 User Identification (UserID)

The UserID is a UTF-8 string value that identifies the user requesting the service. As defined earlier the SubscriberID defines the actual device requesting the service, but there may be multiple users that share that device. The UserID field allows the elements that manage the QoS to associate a request with a specific user.

6.1.14 Shared Resource Identification (SharedResourceID)

The SharedResourceID is only applicable to Multicast Gates and is locally generated by the CMTS. When a Multicast Gate request is successfully processed by the CMTS, the CMTS MUST return a SharedResourceID in the Gate-Set-Ack to the Policy Server.

The CMTS MUST NOT include a SharedResourceID in Gate-Set-Ack messages in response to a Unicast Gate request.

The presence of the SharedResourceID indicates to the PS and AM that resources used in fulfilling the Multicast Gate request are either already shared with a previous Multicast Gate request, or may be shared by future Multicast Gate requests. The SharedResourceID essentially identifies the Multicast Group Service Flow on the DOCSIS network.

The method of allocation of SharedResourceIDs is an implementation option of the CMTS. However, the CMTS MUST ensure the uniqueness of all current SharedResourceID values within the scope of the CMTS. The CMTS SHOULD attempt to minimize the possibility of SharedResourceID ambiguities by ensuring that no SharedResourceID is reused for a different underlying resource within a reasonable duration after the prior closure or deletion of the last gate associated with the SharedResourceID in question.

While each PS may communicate with many CMTSs, a SharedResourceID assigned by one CMTS cannot be guaranteed to be unique across the network. The PS may use additional information, such as the associated CMTS identity, in order to uniquely identify the shared resource.

Note: The SharedResourceID does not replace the GateID, but instead supplements it by providing information about the underlying resource being utilized for a Multicast Gate. The GateID remains the primary and unique identifier of the Gate.

6.2 Gate Transitions

As briefly outlined earlier, a Gate may reside in the following logical states:

- Authorized—a Policy Server has authorized the flow with resource limits defined

- Reserved—resources have been reserved for the flow

- Committed—resources are active and are being used

- Committed-Recovery—inactivity has been detected on the flow; resource recovery pending

For the state machine depicted in Figure 3, the CMTS MUST complete the triggering event with a successful outcome before it transitions a gate from one state to another. For Gate Control events, the CMTS MUST not change state until the request has been fully processed (including any resulting flow transitions) and the CMTS has determined that a success acknowledgement is to be transmitted.
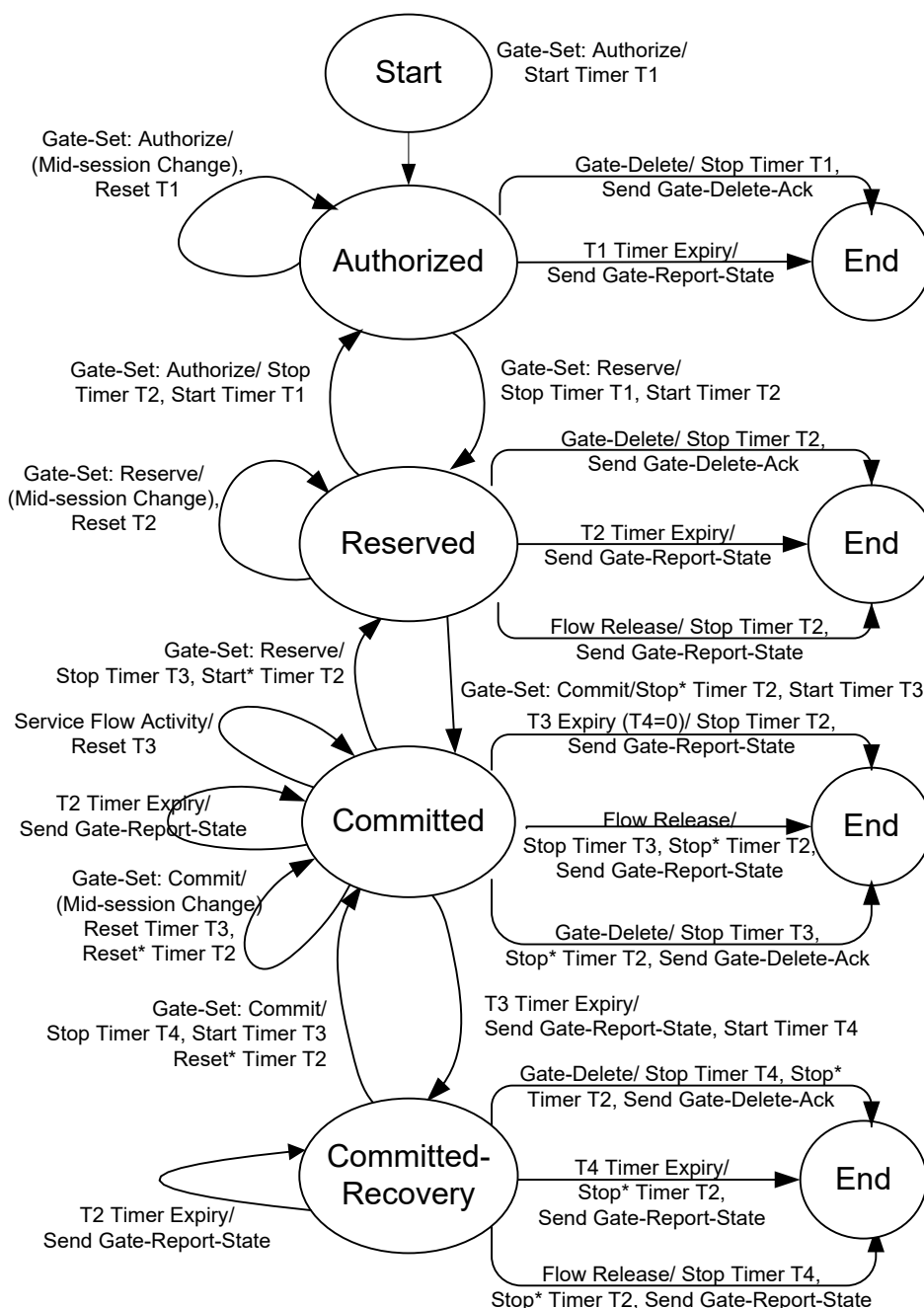
Figure 3 - Gate State Transitions

The CMTS MUST support Gate states and transitions as shown in Figure 3 and described in this section. The CMTS MUST also implement transitions for protocol error processing.

In this section we describe the state transitions of the Gate in the CMTS that result from external events (Gate Control messages from the Policy Server), as well as transitions that result from internal events (e.g., timer expiration). Note that the Policy Server is not the source of the external events; instead the Policy Server is merely acting as a proxy for the Application Manager, which is the trigger for the events.

6.2.1   Authorized

A Gate is created in the CMTS by a Gate-Set command from the Policy Server. The CMTS allocates a unique identifier called a GateID. The Gate is now said to be in the Authorized state and the CMTS MUST start Timer T1. Timer T1 limits the amount of time the authorization remains valid.

A Gate in the Authorized state MUST be deleted upon receipt of a Gate-Delete message. When this happens, the CMTS MUST respond with a Gate-Delete-Ack message and MUST stop Timer T1.

The CMTS is required to support the following state transitions while a Gate is in the Authorized state:

Authorized State Transitions:

- Authorized Loopback to Authorized: Modify Authorized Envelope

- Authorized to Reserved (defines Reserved Envelope ≤ Authorized Envelope)

- Authorized to End (deletes Authorized Envelope)

The CMTS MUST NOT support any other state transitions for a Gate in the Authorized state, but a number of separate stimuli may result in the transitions described.

When the Gate is installed, it is said to be in an Authorized State. While in the Authorized state, the Policy Server MAY modify any of the parameters associated with a Gate (e.g., Traffic Profile, Classifier, etc.). If in an Authorized state a Gate-Set message is received that does not transition the Gate to the Reserved or Committed states, then the CMTS MUST restart timer T1.

While in the Authorized state, the CMTS MUST transition the Gate into the Reserved state upon successful request of the Policy Server. The CMTS MUST transition the Gate into the End state upon receipt of a Gate-Delete from the Policy Server or upon T1 timer expiration.

6.2.2   Reserved

A Gate in the Authorized state is expecting the client to attempt to reserve resources. In Scenario 1, the Policy Server reserves the resources on the client's behalf. To reserve resources, the Policy Server MUST issue a subsequent Gate-Set message with a Traffic Profile that includes the Reserved Envelope. On receipt of this reserve request, the CMTS MUST verify the request is within the authorization limits established for the Gate and perform admission control procedures.

If the Reserve request does not arrive before Timer T1 expires, the CMTS MUST delete the Gate, and notify the Policy Server of the state change. If admission control succeeds and only resource reservation was requested, the CMTS MUST put the Gate in the Reserved state. Simultaneously, the CMTS MUST also stop timer T1 and start timer T2 (Reserved Timer). If admission control procedures are not successful, the CMTS MUST maintain the Gate in the Authorized state and provide a Gate-Set-Err response to the PS.

The CMTS is required to support the following state transitions while a Gate is in the Reserved state:

Reserved State Transitions:

- Reserved Loopback to Reserved: Modify Authorized Envelope (≥ Reserved Envelope)

- Reserved Loopback to Reserved: Modify Reserved Envelope (≤ Authorized Envelope)

- Reserved to Authorized (deletes Reserved Envelope and replaces Authorized Envelope)

- Reserved to Committed (defines Committed Envelope ≤ Reserved Envelope)

- Reserved to End (deletes Reserved and Authorized Envelopes)

The CMTS MUST NOT support any other state transitions for a Gate in the Reserved state, but a number of separate stimuli may result in the transitions described.

From the Authorized state, the CMTS MUST transition the Gate into the Reserved state, if requested by the Policy Server as long as the Reserved envelope is less than or equal to the Authorized envelope, the request passes admission control, and the flow is successfully reserved. Once in the Reserved state, the Gate's Authorized envelope MAY be modified via a Gate-Set message. The Gate's Reserved envelope can also be modified in the Reserved State (see Section 6.5.6). If in a Reserved state a Gate-Set message is received that does not transition the Gate to the Authorized or Committed states, then the CMTS MUST restart timer T2.

If the Commit request does not arrive before timer T2 expires, the CMTS MUST delete the Gate, and notify the Policy Server of the state change.

The Reserved envelope MUST always be less than or equal to the Authorized envelope. In the Reserved state, for a CMTS to transition a Gate to the Committed state, the Committed Envelope MUST be less than or equal to the Reserved Envelope (see Section 6.5.3).

While in the reserved state, the Policy Server may modify the Authorized envelope by specifying a new Traffic Profile in a Gate-Set message. The new Traffic Profile will define a modified Authorized envelope, and the same Reserved Envelope that was used previously to transition the Gate into the Reserved state. However, all requests to modify Authorized, Reserved, or Committed envelopes MUST conform to the general rule:

$$\text{Authorized Envelope} \geq \text{Reserved Envelope} \geq \text{Committed Envelope}$$

The Policy Server MAY delete a Gate in the Reserved state by issuing a Gate-Delete message.

6.2.3 Committed

In the Reserved state the Gate is expecting the client to commit resources, and thereby activate them. In Scenario 1, the Policy Server commits the resources on the client's behalf. To commit resources, the Policy Server MUST issue a Gate-Set command with a Traffic Profile that includes the Committed Envelope. The CMTS MUST authorize the requested Committed envelope against the Reserved envelope, assuming the Reserved and Authorized envelopes remain unchanged. If the authorization succeeds, the CMTS MUST start timer T3, and either stop timer T2 if the Reserved envelope equals the Committed envelope or restart timer T2 if the Reserved Envelope is greater than the Committed envelope. If the authorization fails, the CMTS MUST leave the state of T2 timer unmodified.

Note that once the DOCSIS Service Flow has been activated, the CMTS MUST refresh timer T3 when data is transferred over the flow. If there is no activity on the flow for time equal to T3, the CMTS MUST notify the Policy Server of the state change. Likewise, the Policy Server MUST notify the Application Manager of the state change.

In the Committed state, the Application Manager MAY delete the Gate by issuing a Gate-Delete message to the Policy Server, which in turn MUST relay the message onto the CMTS. If the Policy Server issues a Gate-Delete message to the CMTS, the CMTS MUST delete the Gate and the corresponding Service Flow, and stop timers T2 and T3 if they are running.

The CMTS is required to support the following state transitions while a Gate is in the Committed state:

Committed State Transitions:

- Committed Loopback to Committed: Modify Authorized Envelope ($\geq$ Reserved Envelope)

- Committed Loopback to Committed: Modify Reserved Envelop ($\geq$ Committed Envelope and $\leq$ Authorized Envelope)

- Committed Loopback to Committed: Modify Committed Envelop ($\leq$ Reserved Envelope)

- Committed to Reserved (deletes Committed Envelope)

- Committed to Committed-Recovery: (initiate resource recovery process)

- Committed to End (deletes Committed, Reserved and Authorized Envelopes)

The CMTS MUST NOT support any other state transitions for a Gate in the Committed state, but a number of separate stimuli may result in the transitions described.

While in the Reserved State, the CMTS MUST transition a Gate to the Committed state, if requested by the Policy Server as long as the Committed Envelope is less than or equal to the Reserved Envelope (see Section 6.5.3). While in the Committed state, the Policy Server MAY modify the Authorized Envelope of the Gate via a Gate-Set message, as long as the Authorized Envelope is greater than or equal to the Reserved Envelope. In this state, the Policy Server MAY also modify the Reserved envelope, as long as the reserved envelope is greater than or equal to the Committed Envelope. In this state, the Policy Server MAY even modify the Committed envelope, as long as the new envelope is less than or equal to the Reserved Envelope. If a request is received to de-commit all resources (but keep them reserved) while the Gate is in the Committed state, the CMTS MUST stop timer T3, (re-)start timer T2, and transition back to the Reserved state. In Scenario 1, the Policy Server MAY request this action by issuing a Gate-Set message with the a Traffic Profile that includes the Authorized and Reserved Envelopes, but does not include a Committed Envelope.

While in the Committed state, the CMTS MUST transition a Gate to the End state upon receiving a Gate-Delete message from the Policy Server. While in the Committed state, the Policy Server MAY modify the Authorized or Reserved envelope by simply specifying the new Traffic Profile; the new Traffic Profile MUST contain modified Authorized or Reserved envelopes, and the same Committed Envelope that was used previously to transition the Gate into the Committed state.

When (re-)entering the Committed state due to the receipt of a Gate-Set message, the CMTS MUST (re-)start timer T2 if the Reserved Envelope is greater than the Committed Envelope, or MUST stop timer T2 if it was previously running and the Reserved Envelope is now equal to the Committed Envelope.

While in the Committed state: 1) If the timer T2 expires the Gate MUST remain in the Committed state, the T3 timer MUST be reset, and the CMTS MUST send a Gate-Report-State with Reason code 9 (Gate state unchanged, but T2 timer expiration caused reservation reduction) to the Policy Server indicating the reduction in reserved resources. 2) If a Gate-Set transitions the Gate back to the Reserved state the timer T2 MUST be started if it was not running and restarted if had previously been running, and 3) If a transition to the End state occurs both the timer T2 and T3 MUST be stopped if they are running.

As an optimization, for Scenario 1, the Policy Server MAY Authorize, Reserve, and Commit at the same time by issuing a Gate-Set message with the Traffic Profile that includes all three envelopes set such that the CMTS is told to execute all three actions sequentially without any further interaction from the Policy Server—that is, they must all succeed (if so, the CMTS MUST indicate this by a Gate-Set-Ack) or fail (if so, the CMTS MUST indicate this by a Gate-Set-Err).

From the Committed state, the Gate transition to the Committed-Recovery state due to the expiration of the T3 timer. If the CMTS detects that there has been no activity on the associated flow for a duration of T3, the CMTS MUST start the T4 timer, generate a Gate-Report-State message to the Policy Server indicating that the flow has been inactive for time duration defined by T3 and transition to the Committed-Recovery state, while leaving the associated flow in the activated state. The Policy Server MUST relay the Gate-Report-State message to the Application Manager. The Application Manager MUST either refresh the policy by issuing a Gate-Set message, or remove the Gate by issuing a Gate-Delete message.

Certain applications may not wish to be notified when activity on the flow ceases. In this case, the Application Manager may set the T3 timer (which corresponds to the DOCSIS Active Timer) to 0. As specified in [DOCSIS MULPIv3.0], a value of 0 for the DOCSIS Active Timer indicates that activity detection on the CMTS is turned off for that flow. Thus, the Gate will remain in the Committed state forever until either a Gate-Delete is received or the CM goes offline.

6.2.4   Committed-Recovery

In the Committed state the flow associated with the Gate is active. If the CMTS detects that the flow is unused for a time in excess of the T3 timer, the CMTS notifies the PS (who notifies the AM) that the service-flow associated with the gate has been unused, starts the T4 timer and transitions the Gate to the Committed-Recovery state. (Note: If the T2 timer is running it will continue running. It MUST NOT be reset.) The AM must decide to either refresh the policy by issuing a Gate-Set message to the PS or delete the Gate by issuing a Gate-Delete message to the PS. The Policy Server MUST forward the Gate-Set message or Gate-Delete message to the CMTS.

If, while in the Committed-Recovery state, the CMTS receives a Gate-Set message for the Gate before the timer T4 expires, then the CMTS MUST stop the T4 timer, restart the T3 timer, transition the Gate back to the Committed

state and (re-)start the T2 timer if the Reserved Envelope is greater than the Committed Envelope or stop the T2 timer if it is running and the new Reserved Envelope is equal to the Committed Envelope. If the authorization fails, the CMTS MUST leave the state of the T4 and T2 timers unmodified.

If, while in the Committed-Recovery state, the CMTS receives a Gate-Delete message before the timer T4 expires, then the CMTS MUST stop the T4 Timer, delete the Gate and the corresponding service flow, and stop the T2 timer if it is running.

If the timer T4 expires while in the Committed-Recovery state, then the CMTS MUST send a Gate-Report-State message to the PS, stop the T2 timer if it is running, tear down the service flow associated with the Gate, and then delete the Gate. Likewise, the Policy Server MUST notify the Application Manager of the state change.

If the timer T2 expires while in the Committed-Recovery state, then the Gate MUST remain in the Committed-Recovery state and the CMTS MUST send a Gate-Report-State with Reason code 9 (Gate state unchanged, but T2 timer expiration caused reservation reduction) to the Policy Server indicating the reduction in reserved resources.

The CMTS is required to support the following state transitions while a Gate is in the Committed-Recovery state:

Committed-Recovery State Transitions:

- Committed-Recovery to Committed: (policy has been refreshed)

- Committed-Recovery to End (deletes Committed, Reserved and Authorized Envelopes)

- Committed-Recovery to Committed-Recovery (Timer T2 Expiry; send Gate-Report-State)

The CMTS MUST NOT support any other state transitions for a Gate in the Committed-Recovery state, but a number of separate stimuli may result in the transitions described.

There may be applications that do not wish to keep a Gate once inactivity has been detected. In this instance, the AM may set the T4 timer to zero. When T4 is set to zero, the Committed-Recovery state is bypassed. In this case, when the T3 timer expires the CMTS MUST send a Gate-Report-State message to the PS (with reason code = 5), stop the T2 timer if it is running, tear down the service flow associated with the Gate, and then delete the Gate. Likewise, the Policy Server MUST notify the Application Manager of the state change.

6.3 COPS Profile for PacketCable Multimedia

As defined earlier, admission control involves the process of managing QoS resource requests based on administrative policies and available resources. High-level operational modules associated with this process are depicted in [MM-ARCH]. Under this model administrative policies are stored in a policy database and controlled by the Policy Server.
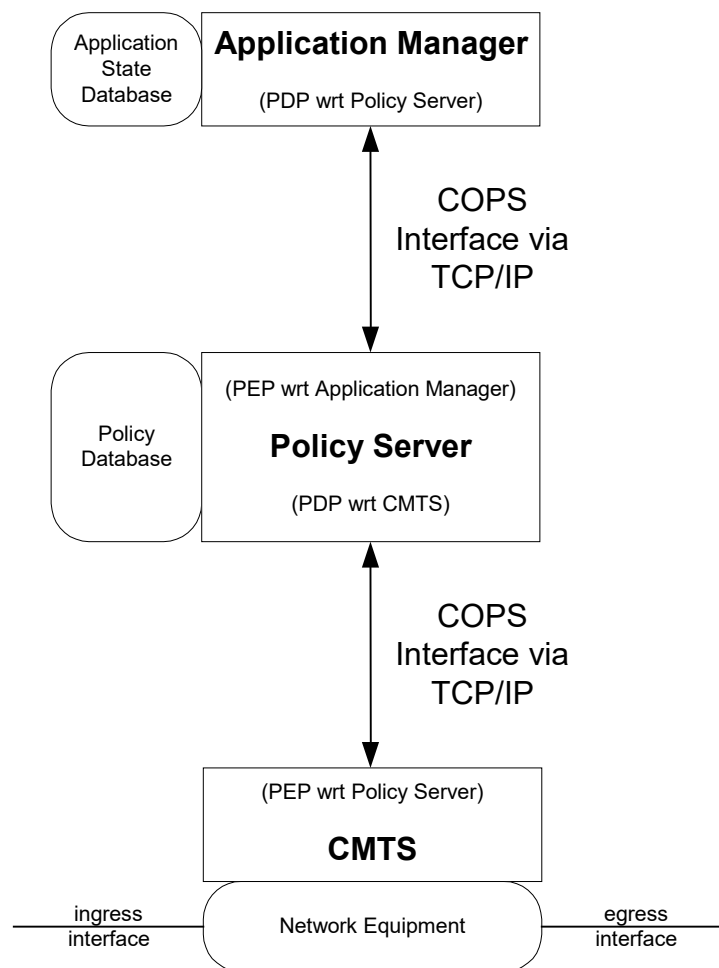
Figure 4 - QoS Admission Control Layout

Admission control decisions made by the Policy Server MUST be communicated to the CMTS or Application Manager using COPS. The CMTS MAY make QoS Admission Control requests to the COPS Server based on network events triggered by either the QoS signaling protocol, or via data flow detection mechanisms. The network event can also be in need of QoS bandwidth management, e.g., a new QoS capable interface becomes operational.

QoS policy decisions made by the Policy Server MAY be pushed to the CMTS based on a request from the Application Manager. The CMTS MAY access that decision information to make policy enforcement decisions on incoming session requests received at the CMTS. The CMTS MUST NOT support CM-initiated DSx messages in PacketCable Multimedia. The CMTS MUST treat a CM-initiated DSx message as a request with an invalid GateID.

A COPS client/server configuration supporting QoS Admission Control is specified in the IETF's COPS protocol [IETF RFC 2748]. This protocol includes the following operations:

- Client-Open (OPN)/Client-Accept (CAT)/Client-Close (CC). The COPS client (PEP) sends an OPN message to initiate a connection with the COPS server (PDP), and the server responds with a CAT message to accept the connection. The server or client sends a CC message to terminate the connection.

- Request (REQ). The client sends a REQ message to the server to request admission control decision information or device configuration information. The REQ message may contain client-specific information that the server uses, together with data in the session admission policy database, to make policy-based decisions.

- Decision (DEC). The server responds to REQs by sending a DEC back to the client that initiated the original request. DEC messages may be sent immediately in response to a REQ (i.e., a solicited DEC) or at any time after to change or update a previous decision (i.e., an unsolicited DEC).

- Report-State (RPT). The client sends a RPT message to the server indicating changes to the request state in the client. The client sends this to inform the server of the actual resource reserved after the server has granted admission. The client can also use Report-State to periodically inform the server the current state of the client.

- Delete-Request-State (DRQ). The client sends a DEL message to the server to request state cleanup. This may be the result of QoS resource release by the client.

- Keep-Alive (KA). Sent by both the client and server for communication failure detection.

Within the PacketCable Multimedia architecture, the PDP-PEP relations are as follows:

- The Application Manager is a COPS Policy Decision Point (PDP) relative to the Policy Server.

- The Policy Server is a PEP relative to the Application Manager.

- The Policy Server is a PDP relative to the CMTS.

- The CMTS is a PEP relative to the Policy Server.

Although the content of COPS messages required for PacketCable Multimedia are consistent with the COPS protocol, there is a slight difference in the way the COPS session starts, and a relaxation of response ordering requirements. [IETF RFC 2748] states:

> "The COPS protocol uses a single persistent TCP connection between the PEP and a remote PDP. One PDP implementation per server MUST listen on a well-known TCP port number (COPS=3288 [IANA]). The PEP is responsible for initiating the TCP connection to a PDP."

The last line of the statement says that the PEP is responsible for initiating the TCP connection. In contrast, in the PacketCable model, the CMTS (PEP) is the one that listens on the assigned port 3918, and it is the Policy Server that MUST initiate the TCP connection to the CMTS. This is the opposite of the model described in the RFC. However, once the TCP connection is in place, the CMTS behaves in a manner consistent with the client, or PEP, in the COPS protocol. Similarly, the Policy Server (PEP) listens on the assigned port 3918 and it is the Application Manager that MUST initiate the TCP connection to the Policy Server.

Note that PacketCable Multimedia and PacketCable 1.x DQoS listen on different ports so that the CMTS may initiate the COPS session with the proper Client-Type.

[IETF RFC 2748] also states:

> "RPT messages solicited by decisions for a given Client Handle MUST set the Solicited Message flag and MUST be sent in the same order as their corresponding Decision messages were received."

The COPS protocol uses the order of RPT and Decision messages to match requests to responses. In contrast, PacketCable Multimedia implementations MUST use the TransactionID object to match responses with requests and SHOULD send RPT messages as soon as they are ready.

[IETF RFC 2748]. This IETF RFC provides a description of the base COPS protocol, independent of Client-Type. The PacketCable architecture is also aligned with the COPS-PR document [IETF RFC 3084]. COPS-PR states:

> "In COPS-PR, policy requests describe the PEP and its configurable parameters (rather than an operational event). If a change occurs in these basic parameters, an updated request is sent. Hence, requests are issued quite infrequently. Decisions are not necessarily mapped directly to requests, and are issued mostly when the PDP responds to external events or PDP events (policy updates)."

When this concept is mapped to the PacketCable Multimedia architecture, the PEP issues a Request to the PDP, specifying a Client-Handle. This Client-Handle is then used in all future Decision messages from the PDP to the PEP. These Decision messages carry the Gate Control messages (i.e., Gate-Set, Gate-Info and Gate-Delete) defined for the DQoS and Multimedia Client-Types. The Client-Handle is used to uniquely identify the PDP-PEP association.

In the PacketCable Multimedia architecture, there can be multiple Application Managers interacting with one or more Policy Servers. There is a single instance of a PacketCable Multimedia COPS session per TCP connection; where a PacketCable Multimedia COPS session refers to the Gate messages between the PDP and PEP associated with a single Client-Handle. This means there is one COPS-TCP connection between an Application Manager and a Policy Server. Similarly, there can be one or more Policy Servers talking to one or more CMTSs. When connected to multiple PDPs, the PEP MUST ensure that the Client-Handle used is unique per association.

6.4   Gate Control Protocol Message Formats

Protocol messages for Gate Control MUST be transported within the COPS protocol messages. The PDP and PEP MUST establish and use a TCP connection for communication, and utilize the mechanisms specified in [PKT-SEC1.5] to secure the communication path.

6.4.1   COPS Common Message Format

Each COPS message consists of the COPS header followed by a number of typed objects. The Application Manager, Policy Server and CMTS MUST use the COPS Common Message format as defined below as the message format for all message exchanges. In the object specifications that follow, each row represents a 4-byte word as all objects align on 4-byte word boundaries.

0		1	2	3
Version	Flags	Op-Code	Client-Type	
Message Length				

Version is a 4-bit field giving the current COPS version number. This field MUST be set to 1.

Flags is a 4-bit field. The least significant bit is the solicited message flag. When a COPS message is sent in response to another message (e.g., a solicited decision sent in response to a request) this flag MUST be set to 1. In other cases (e.g., an unsolicited decision) the flag MUST NOT be set (value = 0). In keeping with the DQoS model, the first Decision message sent in response to a Request message is a solicited response and its solicited message flag MUST be set. All other Decision messages are unsolicited and the solicited message flag MUST be cleared. All other flags MUST be set to zero.

Op-code is a 1-byte unsigned integer field that gives the COPS operation to be performed. COPS operations used in this PacketCable specification are:

    1 = Request (REQ)
    2 = Decision(DEC)
    3 = Report-State (RPT)
    4 = Delete Request State (DRQ)
    6 = Client-Open (OPN)
    7 = Client-Accept (CAT)
    8 = Client-Close (CC)
    9 = Keep-Alive (KA)

Client-Type is a 2-byte unsigned integer identifier. For PacketCable Multimedia use, the Client-Type MUST be set to PacketCable Multimedia client (0x800A). For Keep-Alive messages (Op-code = 9) the Client-Type MUST be set to zero, as the KA is used for connection verification rather than per-client session verification.

Message Length is a 4-byte unsigned integer value giving the size of the overall message in octets. Messages MUST be aligned on 4-byte boundaries, so the length MUST be a multiple of four.

Following the COPS common header are one or more objects. All the objects MUST conform to the same object format where each object consists of one or more 4-byte words with a four-octet header, using the following format.

0	1	2	3
Length		C-Num	C-Type
Object Contents			

Length is a 2-byte unsigned integer value that MUST give the number of bytes (including the header) that compose the object. If the original length in octets is not a multiple of four, padding MUST be added to the end of the object so that it is aligned to the next 4-byte boundary.

C-Num identifies the class of information contained in the object, and the C-Type identifies the subtype or version of the information contained in the object. Standard COPS objects (as defined in [IETF RFC 2748]) used in this specification, and their C-Num values, are:

    1 = Handle
    2 = Context
    6 = Decision
    8 = Error
    9 = Client Specific Info
    10 = Keep-Alive-Timer
    11 = PEP Identification
    12 = Report Type

Each of these objects MUST conform to the format and rules relating to the individual object as defined in [IETF RFC 2748].

6.4.2   Additional COPS Objects for Gate Control

As with the COPS-PR and COPS-RSVP profiles, the PacketCable Client-Type defines a number of additional object formats. These objects MUST be placed inside a Decision object, C-Num = 6, C-Type = 4 (Client Specific Decision Data) when carried from PDP to PEP in a Decision message. They MUST also be placed in a ClientSI object, C-Num = 9, C-Type = 1 (Signaled ClientSI) when carried from the PEP to the PDP in a Report State message or Client-Open message.

These objects are encoded similarly to the client-specific objects for COPS-PR and as in COPS-PR these objects are numbered using a client-specific number space, which is independent of the top-level COPS object number space. For this reason, the object numbers and types are given as S-Num and S-Type, respectively. S-Num and S-Type MUST be one octet. The COPS Length field MUST be two octets. Additional COPS objects are defined for use by PacketCable Multimedia in the following sections.

6.4.2.1   TransactionID

TransactionID is a 2-byte unsigned integer quantity, which contains a token that is used by the Application Manager to match responses from the Policy Server and by the Policy Server to match responses from the CMTS to the previous requests. The TransactionID MUST also contain the command type that identifies the action to be taken or response. The TransactionID Object MUST conform to the following format.

Length = 8	S-Num = 1	S-Type = 1
Transaction Identifier	Gate Command Type	

The Transaction Identifier MUST be set to 0 when included in a Gate-Report-State message.

Gate Command Type is a 2-byte unsigned integer value which identifies the Gate Control message type and MUST be one of the following:

    <Reserved>   1-3
    Gate-Set   4
    Gate-Set-Ack   5
    Gate-Set-Err   6
    Gate-Info   7
    Gate-Info-Ack   8
    Gate-Info-Err   9
    Gate-Delete   10
    Gate-Delete-Ack   11
    Gate-Delete-Err   12
    <Reserved>   13

<Reserved>	14
Gate-Report-State	15
Gate-Cmd-Err	16
PDP-Config	17
PDP-Config-Ack	18
PDP-Config-Err	19
Synch-Request	20
Synch-Report	21
Synch-Complete	22
Msg-Receipt	23

6.4.2.2   AMID

AMID, the Application Manager ID, consists of two fields, the Application Manager Tag and Application Type. The Application Manager Tag is a 2-byte unsigned integer value which identifies the Application Manager responsible for handling the session. The Application Type is a 2-byte unsigned integer value which identifies the type of application that this gate is associated with. The Application Manager MUST include this object in all messages it issues to the Policy Server. The Policy Server MUST include the received AMID in all messages it issues down to the CMTS in response to the messages it receives from the Application Manager. The CMTS MUST include the received AMID object in all messages it issues to the Policy Server, and likewise, the Policy Server MUST include the received AMID to the Application Manager. The Policy Server may use the AMID in messages from the CMTS to resolve the Application Manager to which it may need to generate a message.

Application Type is used by an Application Manager to identify the application that a gate is associated with. There is only one reserved value, zero, which indicates that there is "no defined application association". An Application Manager MAY pass a non-zero value to the Policy Server to indicate the particular application that a gate is associated with. The Policy Server MAY use this value when applying policies to this gate. The CMTS MAY use this value when performing admission control for this gate. The Application Type values are configured by the service provider in the Application Managers and Policy Servers. In order for the CMTS to utilize the Application Type in its admission control logic, the service provider would need to configure the CMTS with the allowed Application Types and their associated meaning. These values are unique within the domain of the service provider, and have context only within this domain. Therefore a single Application Manager that is servicing multiple service providers may be configured with a different Application Type for each service provider, for the same application.

The AMID object MUST conform to the following format.

Length = 8		S-Num = 2	S-Type = 1
Application Type		Application Manager Tag	

6.4.2.3   SubscriberID

There are two different versions of the SubscriberID object which differ in the IP version of the addresses that they contain. The simply named SubscriberID object contains a 4-byte value giving the IPv4 address (represented as four concatenated octet values) of the subscriber for this service request. The IPv6SubscriberID object contains a 16-byte value giving the IPv6 address for this request. This IPv4 or IPv6 address may be the actual IP address of the subscriber CPE device requesting service (if this address is routable and visible from the head-end) or this address may be the IP address of the CM serving this subscriber (if NAT is performed behind the CM). In the message definitions provided elsewhere in this specification the term SubscriberID refers generically to either a SubscriberID object containing an IPv4 address or an IPv6SubscriberID object containing an IPv6 Address. This object is used to route Gate Control messages within a complex network of PS and CMTS elements. It may also be used in the definition and enforcement of per-subscriber policy rules. The same value of SubscriberID provided by the PDP in the initial gate-set message for the gate MUST be used in all subsequent Gate-Set, Gate-Info, and Gate-Delete messages submitted by the PDP for that gate. The SubscriberID object MUST conform to the following format.

Length = 8	S-Num = 3	S-Type = 1
SubscriberID (4-octet IPv4 Address)		

The IPv6SubscriberID object MUST conform to the following format.

Length = 20		S-Num = 3	S-Type = 2
SubscriberID (16-octet IPv6 Address)			

6.4.2.4   GateID

GateID is a 4-byte unsigned integer value which identifies the Gate referenced in the command message, or referenced by the CMTS for a response message. The CMTS MUST ensure the GateID is unique. If the CMTS also supports PacketCable 1.x, the GateID MUST NOT duplicate a PacketCable 1.x GateID currently in use. The GateID object MUST conform to the following format.

Length = 8	S-Num = 4	S-Type = 1
GateID		

6.4.2.5   GateSpec

GateSpec defines a specific set of attributes associated with a Gate. The GateSpec object MUST conform to the following format.

Length = 16		S-Num = 5	S-Type = 1
Flags	DSCP/TOS Overwrite	DSCP/TOS Mask	SessionClassID
Timer T1		Timer T2	
Timer T3		Timer T4	

Flags is a 1-byte bit-field value defined as follows:

- Bit 0: direction bit, MUST be either zero for a downstream Gate, or one for an upstream Gate.

- Bit 1: DSCP/TOS enable bit, MUST be either zero to disable DSCP overwrite, or one to enable.

- Persistence enable bit, MUST be either zero (default) to disable Gate persistence, or one to enable persistence.

- Bits 3-7: reserved, MUST be zero.

The same value of the direction bit provided by the PDP in the initial Gate-Set message MUST be used in all subsequent Gate-Set messages submitted by the PDP for that gate.

If the persistence bit is set, the CMTS MUST save PCMM Gate information when a cable modem goes offline. https://cablelabs.jamacloud.com/perspective.req?docId=486595&projectId=156 When the cable modem comes online, the CMTS MUST recreate any persisted Gates for the cable modem using the saved Gate information. The CMTS MUST identify the cable modem by MAC address and generate a Gate-Report-State message when the Gate is recreated (Reason code 17) returning the possibly updated SubscriberID (IP Address) to allow the Policy Server and Application Manager to re-establish the mapping from GateID to SubscriberID.

When restoring persisted Gates, the CMTS MUST use the previously assigned GateID.

SessionClassID is a 1-byte unsigned integer value which identifies the proper admission control policy or parameters to be applied for this Gate. The SessionClassID is a bit field, defined as follows:

- Bit 0-2: Priority, a number from 0 to 7, where 0 is low priority and 7 is high.

- Bit 3: Preemption, set to enable preemption of bandwidth allocated to lower priority sessions if necessary (if supported).

- Bit 4-7: Configurable, default to 0

The priority field describes the relative importance of the session as compared to other sessions generated by the same PDP. The PEP MAY use this value to both implement priority based admission (in conjunction with the

Preemption bit), and to defend the resulting flow from being preempted (see [IETF RFC 2751] "defending priority"). The priority granularity of a CMTS may be smaller than the 8 available values. In this case the CMTS SHOULD distribute its levels across the entire priority range. For example, if the CMTS defines 2 priority levels, then it should interpret values 0 to 3 as low priority and values 4 to 7 as high priority. The Policy Server SHOULD normalize or otherwise transform this value to ensure a coherent priority system across the operator's CMTSs, but respond to any AM requests with the original priority value.

The preemption bit is used to by a PDP to direct the PEP to apply a priority-based admission control. Preemption support is optional; a PEP MAY ignore this bit. If preemption is not requested by the PDP or not implemented by the PEP, then admission control policy will be on a first-come, first-served basis. If the PEP decides to preempt, it MUST preempt bandwidth from sessions whose priority is lower than this session's priority, beginning with the lowest priority session(s). This bit is not used to control which sessions are preemptable; instead an unpreemptable session is requested by using the highest priority. If a lower priority session is terminated as a result of preemption, the PEP MUST send a Gate-Report-State message to the PDP with a GateState object Reason field of 1—"Close initiated by CMTS due to reservation reassignment" and transition the gate to the "end" state.

Application Managers that provide novel services MAY use the Configurable field to specify new session classes. The Policy Server MAY support configurable policies based on this value and MAY rewrite this field before forwarding the message to the CMTS. A CMTS MAY implement a novel session class metric using these bits, but the value 0 MUST map to a reasonable default for a PDP that is uninterested in this metric.

The DSCP/TOS Overwrite is a 1-byte bit field [IETF RFC 2474] defined by the following alternative structures, depending upon network management strategy. This field, combined with the 1-byte DSCP/TOS Mask, is used to identify particular bits within the IPv4 DSCP/TOS byte.

0	1	2	3	4	5	6	7
Differentiated Services Code Point (DSCP)						Not Used	Not Used

0	1	2	3	4	5	6	7
IP Precedence			IP TOS				Not Used

If the 'Enable' bit in the GateSpec Flags field is set, then the CMTS MUST mark the packets traversing the CMTS DSCP/TOS value. If the 'Enable' bit is cleared, then the CMTS MUST NOT perform any marking.

When converting the DSCP/TOS Overwrite and the DSCP/TOS Mask from the GateSpec to the corresponding DOCSIS Service Flow Parameters, the CMTS MUST translate the values in the following way:

DOCSIS tos-and-mask = ~ DSCP/TOS Mask

DOCSIS tos-or-mask = DSCP/TOS Mask AND DSCP/TOS Overwrite

Timers T1, T2, T3, and T4 are 2-byte unsigned integers specified in seconds, and MUST be used as outlined in the Gate Transition Diagram as described in Section 6.2. A value of zero for T1 indicates that the CMTS provisioned value for the timer MUST be used. T2 corresponds to the DOCSIS Admitted timer and T3 corresponds to the DOCSIS Active timer. All corresponding DOCSIS requirements apply to these timers. Specifically, a zero value for either of these timers indicates that the corresponding timer MUST be disabled.

6.4.2.6   *Classifiers*

There are three types of classifiers supported in this specification. An Application Manager MUST use the Classifier, Extended Classifier, and IPv6 Classifier objects in accordance with Section 6.1.5.

The Classifier object specifies the packet matching rules associated with a Gate. As defined in Sections 6.4.3.1 and 6.4.3.2, for Unicast Gates multiple Classifier objects may be included in the Gate-Set to allow for complex classifier rules. When an AM is using Classifier objects, at least one Classifier MUST be provided by the PDP in all Gate-Set messages. More than one Classifier is allowed for Unicast Gates. Only one classifier is required to be supported for Multicast Gates. Unlike DOCSIS DSx signaling, Classifiers have no identifier and have no explicit add, change, or remove operations. The entire set of Classifiers present in a Gate-Set message replaces the entire set of classifiers for the existing Gate. Equivalence of Classifiers is determined by comparing all fields within the Classifier. Classifiers may be provided in any order.

The Classifier object MUST conform to the following format.

Length = 24		S-Num = 6	S-Type = 1
Protocol ID		DSCP/TOS Field	DSCP/TOS Mask
Source IP Address (4-octets)			
Destination IP Address (4-octets)			
Source Port		Destination Port	
Priority	Reserved		

The Extended Classifier object also specifies the packet matching rules associated with a Gate, but includes more detailed information for matching traffic, as well adding, modifying, deleting, activating and inactivating Classifiers. As defined in Sections 6.4.3.1 and 6.4.3.2, for Unicast Gates multiple Extended Classifier objects may be included in the Gate-Set to allow for complex classifier rules. However, since the ordering of objects in a message and the order of processing those objects is not mandated, An Application Manager SHOULD NOT send a GateSet with multiple Extended Classifiers with the same ClassifierID, yet different Actions. When an AM is using Extended Classifier objects, at least one Extended Classifier MUST be provided by the PDP in all Gate-Set messages. For Unicast Gates, more than one Extended Classifier is allowed. For Multicast Gates, only one Extended Classifier is required to be supported. Since the Extended Classifier is based on the DOCSIS IP Classifier, all DOCSIS classifier semantics apply, with the exception that at least one Extended Classifier be present in a Gate-Set message.

The Extended Classifier object MUST conform to the following format.

Length = 40		S-Num = 6	S-Type = 2
Protocol ID		DSCP/TOS Field	DSCP/TOS Mask
IP Source Address (4-octets)			
IP Source Mask (4-octets)			
IP Destination Address (4-octets)			
IP Destination Mask (4-octets)			
Source Port Start		Source Port End	
Destination Port Start		Destination Port End	
ClassifierID		Priority	Activation State
Action	Reserved		

IP Source Address is a 4-octet IPv4 address, conforming to section C.2.1.6.3 of [DOCSIS MULPIv3.0], or zero for no match (i.e., a wildcard specification that will match any packet).

For the Extended Classifier, the IP Source Mask is 4-octet IPv4 mask, conforming to section C.2.1.6.4 of [DOCSIS MULPIv3.0]. When the IP Source Address is zero, the IP Source Mask is irrelevant. When the IP Source Address is non-zero, only packets with source IP address 'pkt.ip-src' will match if (pkt.ip-src AND classifier.ipmask-src) == classifier.ip-src.

IP Destination Address is a 4-octet IPv4 address, conforming to section C.2.1.6.5 of [DOCSIS MULPIv3.0], or zero for no match (i.e., a wildcard specification that will match any packet).

For the Extended Classifier, the IP Destination Mask is 4-octet IPv4 mask, conforming to section C.2.1.6.6 of [DOCSIS MULPIv3.0]. When the IP Destination Address is zero, the IP Destination Mask is irrelevant. When the IP Destination Address is non-zero, only packets with destination IP address 'pkt.ip-dst' will match if (pkt.ip-dst AND classifier.ipmask-dst) == classifier.ip-dst.

For the Classifier object, Source Port and Destination Port MUST be a pair of 2-byte unsigned integer values, or zero for no match.

For the Extended Classifier object, the Source Port Start specifies the low-end TCP/UDP source port value and conforms to section C.2.1.7.1 of 0[DOCSIS MULPIv3.0]. Source Port End specifies the high-end TCP/UDP source port value and conforms to section C.2.1.7.2 of [DOCSIS MULPIv3.0]. An IP packet with TCP/UDP source port value "src-port" matches if Source Port Start ≤ src-port ≤ Source Port End. Likewise, the Destination Port Start

specifies the low-end TCP/UDP destination port value and conforms to section C.2.1.7.3 of [DOCSIS MULPIv3.0]. Destination Port End specifies the high-end TCP/UDP destination port value and conforms to section C.2.1.7.4 of [DOCSIS MULPIv3.0]. An IP packet with TCP/UDP destination port value "dst-port" matches if Destination Port Start ≤ dst-port ≤ Destination Port End. When using the Source Port or Destination Port ranges, a Port Start = 0 and Port End = 65535 represents a wildcarded port classification.

Protocol ID MUST conform to section C.2.1.6.2 of [DOCSIS MULPIv3.0], or zero for no match.

DSCP/TOS Field is a 1-byte bit field which MUST conform to the following alternative structures:

0	1	2	3	4	5	6	7
Differentiated Services Code Point (DSCP)						Not Used	Enable

0	1	2	3	4	5	6	7
IP Precedence			IP TOS				Enable

DSCP/TOP Mask is a 1-byte bit field providing a bit mask used to select relevant bits from the accompanying DSCP/TOP Field value.

If the 'Enable' bit is set, then the CMTS MUST use these values to construct the IP TOS Range and Mask field specified in its DSx messaging. If the 'Enable' bit is cleared, then the CMTS MUST omit the IP TOS Range and Mask values from its DSx messaging and exclude the IP TOS byte from the packet classification process.

The IPv6 Classifier object also specifies the packet matching rules associated with a Gate, when IPv6 Addresses are used. As defined in Sections 6.4.3.1 and 6.4.3.2, for Unicast Gates multiple IPv6 Classifier objects may be included in the Gate-Set to allow for complex classifier rules. However, since the ordering of objects in a message and the order of processing those objects is not mandated, an AM SHOULD NOT send a GateSet with multiple IPv6 Classifiers with the same ClassificationID, yet different Actions. When an AM is using IPv6 Classifier objects, at least one IPv6 Classifier MUST be provided by the PDP in all Gate-Set messages. For Unicast Gates more than one IPv6 Classifier is allowed. For Multicast Gates only one IPv6 Classifier is required to be supported. Since the IPv6 Classifier is based on the DOCSIS IPv6 Classifier, all DOCSIS classifier semantics apply, with the exception that at least one IPv6 Classifier be present in a Gate-Set message.

The IPv6 Classifier object MUST conform to the following format.

Length = 64			S-Num = 6	S-Type = 3	
Reserved	Flags	tc-low	tc-high	tc-mask	
Flow Label					
Next Header Type			Source Prefix Length	Destination Prefix Length	
IPv6 Source Address (16-octets)					
IPv6 Destination Address (16-octets)					
Source Port Start			Source Port End		
Destination Port Start			Destination Port End		
ClassifierID			Priority	Activation State	
Action		Reserved			

For the IPv6 Classifier, the Traffic Class Range and Mask fields, tc-low, tc-high, and tc-mask, allow matching on the IPv6 Traffic Class value. An IPv6 packet with IPv6 Traffic Class value "ip-tc" matches this parameter if tc-low ≤ (ip-tc AND tc-mask) ≤ tc-high. If the tc-mask field is set to 0, then comparison of the IPv6 packet Traffic Class byte for this entry is irrelevant.

Note: The value 0x3F for tc-mask will exclude the Explicit Congestion Notification on [DOCSIS MULPIv3.0] bits from the comparison, and hence will result in classification based on DSCP [IETF RFC 2474]. For further discussion of the IPv6 Traffic Class Range and Mask fields, refer to C.2.1.10.1 of [DOCSIS MULPIv3.0].

For the IPv6 Classifier, Flags is a 4-bit field. The least significant bit is the Flow Label flag. When the Flow Label field contains valid data for comparison with the IPv6 Flow Label, this flag MUST be set to 1. When comparison of the IPv6 Flow Label for this entry is irrelevant then the Flow Label flag MUST NOT be set (value = 0). When the Flow Label flag is set to 0, the CMTS MUST NOT include the IPv6 Flow Label field in the classifier. All other flags MUST be set to zero.

For the IPv6 Classifier, the Flow Label field matches the Flow Label used in the IPv6 header. The 20 least significant bits represent the 20-bit IPv6 Flow Label while the 12 most significant bits are ignored. For further discussion of the IPv6 Flow Label field, refer to C.2.1.10.2 of [DOCSIS MULPIv3.0].

For the IPv6 Classifier, the value of the Next Header Type field specifies the desired next header type value for any header or extension header associated with the packet. Typically this value will specify the next layer protocol type. There are two special IPv6 next header type field values: "256" matches traffic with any IPv6 next header type value, and "257" matches both TCP and UDP traffic. Values greater than 257 are invalid for comparisons (i.e., no traffic can match this entry). For further discussion of the IPv6 Next Header Type field, refer to C.2.1.10.3 of [DOCSIS MULPIv3.0].

For the IPv6 Classifier, the value of the Source Prefix Length field specifies the fixed, most significant bits of an IPv6 address that are used to determine address range and subnet ID for the IPv6 Source Address. For further discussion of the IPv6 Source Prefix Length field, refer to C.2.1.10.5 of [DOCSIS MULPIv3.0]

For the IPv6 Classifier, the value of the Destination Prefix Length field specifies the fixed, most significant bits of an IPv6 address that are used to determine address range and subnet ID for the IPv6 Destination Address. For further discussion of the IPv6 Destination Prefix Length field, refer to C.2.1.10.7 of [DOCSIS MULPIv3.0].

For the IPv6 Classifier, the value of the IPv6 Source Address field specifies the matching value for the IPv6 source address. An IPv6 packet with IPv6 source address "ip6-src" matches this parameter if src = (ip6-src AND smask). "smask" is computed by setting the most significant 'n' bits of smask to 1, where 'n' is IPv6 Source Prefix Length in bits. If the IPv6 Source Address parameter is set to 0, then comparison of the IPv6 packet source address for this entry is irrelevant. For further discussion of the IPv6 Source Address field, refer to C.2.1.10.4 of [DOCSIS MULPIv3.0].

For the IPv6 Classifier, the value of the IPv6 Destination Address field specifies the matching value for the IPv6 destination address. An IPv6 packet with IPv6 destination address "ip6-dst" matches this parameter if dst = (ip6-dst AND dmask). "dmask" is computed by setting the most significant 'n' bits of dmask to 1, where 'n' is IPv6 Destination Prefix Length in bits. If the IPv6 Destination Address parameter is set to 0, then comparison of the IPv6 packet destination address for this entry is irrelevant. For further discussion of the IPv6 Destination Address field, refer to C.2.1.10.6 of [DOCSIS MULPIv3.0].

Priority is a 1-byte field that allows differentiation between classifiers that might overlap. A default value of 64 SHOULD be used if a specific priority value is not required. For further discussion of the Priority field, refer to C.2.1.4.5of [DOCSIS MULPIv3.0].

For the Extended Classifier or the IPv6, the ClassifierID is a 2 byte unsigned integer which is used as an identifier for the Extended Classifier. This value MUST be unique per Gate. The Application Manager assigns the ClassifierID.

For the Extended Classifier or the IPv6, the Activation State is a 1 byte unsigned integer with the following values:

0x00   Inactive
0x01   Active

All other values are reserved. If a value other than 0 or 1 is used by the AM, the CMTS MUST generate a Gate-Set-Err with PacketCable Error 17 (Invalid Field Value in Object) and Error-Subcode of 6 | 2 (S-Num | S-Type). When the Activation State is set to 0 (Inactive) the classifier MUST NOT be used to match traffic. When set to 1 (Active), the classifier MUST be used to match traffic.

For the Extended Classifier or the IPv6, the Action field is a 1 byte unsigned integer with the following values:

0x00	Add classifier
0x01	Replace classifier
0x02	Delete classifier
0x03	No change

All other values are reserved. If a value other than 0, 1, 2, or 3 is used by the AM, the CMTS MUST generate a Gate-Set-Err, with PacketCable Error 17 (Invalid Field Value in Object) and Error-Subcode of 6 | 2 (S-Num | S-Type). The Action field is meant to assist the CMTS with managing the list of classifiers associated with a Gate.

When the Action is set to 0 (Add classifier) by the AM, the classifier will be added to the list of classifiers for the Gate. If the Action field is set to 0 (Add classifier) for a ClassifierID currently being used for this Gate, the CMTS MUST generate a Gate-Set-Err with PacketCable Error 17 (Invalid Field Value in Object) and Error-Subcode of 6 | 2 (S-Num | S-Type).

When set to 1 (Replace classifier) by the AM, the classifier matching the ClassifierID specified will be replaced with the information in the Extended Classifier object. If the Action field is set to 1 (Replace classifier) with a ClassifierID not known to the CMTS, the CMTS MUST generate a Gate-Set-Err with PacketCable Error 17 (Invalid Field Value in Object) and Error-Subcode of 6 | 2 (S-Num | S-Type).

When set to 2 (Delete classifier) by the AM, the classifier matching the ClassifierID will be removed from the list of classifiers for the Gate. If the Action field is set to 2 (Delete classifier) with a ClassifierID not known to the CMTS, the CMTS MUST generate a Gate-Set-Err with PacketCable Error 17 (Invalid Field Value in Object) and Error-Subcode of 6 | 2 (S-Num | S-Type).

When set to 3 (No Change) by the AM, the CMTS will leave the classifier matching the ClassifierID unmodified. This action is technically the same as not sending the Extended Classifier for the ClassifierID at all, however, this method MUST be used when the AM only has a single classifier for a Gate and needs to modify the Gate. This is due to the fact that every Gate-Set must contain at least one classifier. If the Action field is set to 3 (No Change) with a ClassifierID not known to the CMTS, the CMTS MUST generate a Gate-Set-Err with PacketCable Error 17 (Invalid Field Value in Object) and Error-Subcode of 6 | 2 (S-Num | S-Type).

The CMTS MUST only use the value of 3 (No change) in a Gate-Info-Ack, and is considered the default value of this field.

6.4.2.7 Traffic Profiles

There are four different ways to express a traffic profile. The traffic profile can be expressed via a FlowSpec, a DOCSIS Service Class Name, DOCSIS-specific parameters or Upstream Drop. The four methods are distinguished via a different S-Type value in the Traffic Profile (S-Num = 7) object. S-Type of 1 indicates the object contains a traffic profile specified in RSVP FlowSpec format. S-Type of 2 indicates the object contains a traffic profile specified in DOCSIS Service Class Name format. S-Type of 3–8 indicates the object contains a traffic profile that is specified via DOCSIS-specific parameters. S-Type of 9 indicates the object contains a traffic profile specified in an Upstream Drop format.

All Traffic Profiles utilize "replace" semantics, meaning that the envelopes present in this Traffic Profile replace all existing envelopes associated with the Gate and corresponding Service Flow. Thus, all traffic parameters associated with a given Gate MUST be included in every message that includes a Traffic Profile. The traffic profile format (RSVP FlowSpec, DOCSIS Service Class Name, DOCSIS-specific parameters, Upstream Drop) for a specific envelope MUST remain constant and unchanged throughout the life of the gate.

All Traffic Profiles share a common field known as the Envelope Field. This field is a bit field that signals the envelope types (i.e., Authorized, Reserved, and Committed) that are present in the object. A value of 1 in a given bit field indicates that the envelope type is present in the Traffic Profile.

- Bit 0: Authorized Envelope

- Bit 1: Reserved Envelope

- Bit 2: Committed Envelope

Thus a bit pattern of 001 (or 0x01) indicates the presence of only the Authorized Envelope, while a value of 111 (or 0x7) indicates the presence of all three envelopes. Only the following values are legal: 001, 011 and 111; the Envelope Field MUST be set to one of these three legal values. Further limitations on the value of the Envelope Field may be a function of the current state of the Gate. Refer to Section 6.2 for more information.

For the Traffic Profile formats that allow multiple sets of envelope parameters, the mapping of envelope parameter sets follows one of the following methods:

- If all of the envelope types that are indicated in the envelope field share a common set of envelope parameters, then the PDP SHOULD ensure that exactly one set of envelope parameters are present in the traffic profile. This allows for the most efficient transmission and processing of the Traffic Profile throughout the system.

- Otherwise, the PDP MUST ensure that exactly one set of envelope parameters is included for each of the envelope types that are indicated in the envelope field. The proper order of the envelope parameter sets is shown in the appropriate message diagram in Sections 6.4.2.1 and 6.4.2.3 thru 6.4.2.7.8.

While all Traffic Profiles end up providing QoS on the access network, it is important to note several subtle differences between the signaling mechanisms. As noted previously, the conversion of a FlowSpec (S-Type 1) to DOCSIS parameters by the CMTS is generally less efficient than specifying the DOCSIS parameters themselves. That said, specifying DOCSIS parameters explicitly (S-Types 3-7) is not a panacea either, the QoS MIB only logs QoS information about named Service Flows in its ServiceFlowLogTable. Thus, only flows created via S-Type 2 will have logged QoS information in this table. For some this may not be a major issue, but for debugging and just general operational tracking this subtlety should be taken into account by operators and Application Manager vendors evaluating the Traffic Profile signaling alternatives provided by this specification.

Note: Based on the evolution of the service flow parameters in the latest DOCSIS specifications [DOCSIS MULPIv3.1] [DOCSIS MULPIv4.0], updates were made in this specification, to the Best Effort Service and Downstream service traffic profiles and also added a new traffic profile, Proactive Grant service. No updates were chosen to be made to the other traffic profiles: NRTPS, RTPS, UGA, UGS-AD, as there were not any new use cases which needed the changes to the parameters for those services.

6.4.2.7.1   Flow Spec

The FlowSpec object defines the Traffic Profile associated with a Gate through an RSVP-like parameterization scheme. The mapping of these parameters to DOCSIS parameters is specified in Section 8. The FlowSpec object MUST conform to the following specification:

Length = 36 or 64 or 92		S-Num = 7	S-Type = 1
Envelope	Service Number	Reserved	Reserved
Authorized Envelope			
Token Bucket Rate [r] (IEEE floating point number)			
Token Bucket Size [b] (IEEE floating point number)			
Peak Data Rate (p) (IEEE floating point number)			
Minimum Policed Unit [m] (integer)			
Maximum Packet Size [M] (integer)			
Rate [R] (IEEE floating point number)			
Slack Term [S] (integer)			
Reserved Envelope (optional)			
Token Bucket Rate [r] (IEEE floating point number)			

Token Bucket Size [b] (IEEE floating point number)	
Peak Data Rate (p) (IEEE floating point number)	
Minimum Policed Unit [m] (integer)	
Maximum Packet Size [M] (integer)	
Rate [R] (IEEE floating point number)	
Slack Term [S] (integer)	
Committed Envelope (optional)	
Token Bucket Rate [r] (IEEE floating point number)	
Token Bucket Size [b] (IEEE floating point number)	
Peak Data Rate (p) (IEEE floating point number)	
Minimum Policed Unit [m] (integer)	
Maximum Packet Size [M] (integer)	
Rate [R] (IEEE floating point number)	
Slack Term [S] (integer)	

The Service Number field corresponds to the RSVP FlowSpec service number as defined in [IETF RFC 2205]. If Service Number is set to five, this indicates Controlled Load service and the CMTS MUST utilize only the TSpec values (i.e., token bucket parameters) to perform the necessary authorization, reservation and commit operations. For Controlled Load service, the CMTS MUST ignore the RSpec R and S fields.

If Service Number is set to two, this signals Guaranteed service and the CMTS MUST utilize both the TSpec and RSpec values to perform the necessary authorization, reservation and commit operations.

The values r, b, p, m, M, R, and s are defined and described in Section 9.

6.4.2.7.2   DOCSIS Service Class Name

The DOCSIS Service Class Name object defines the preconfigured Service Class Name associated with a Gate. The DOCSIS Service Class Name object MUST conform to the following specification:

Length = 12 or 16 or 20 or 24		S-Num = 7	S-Type = 2
Envelope	Reserved	Reserved	Reserved
Service Class Name			

The Service Class Name is MUST be 2-16 bytes of null-terminated ASCII string. (Refer to section C.2.2.3.4 of [DOCSIS MULPIv3.0]) This name MUST be padded with null bytes to align on a 4-byte boundary.

Note that unlike a FlowSpec Traffic Profile which allows for different parameters to be associated with each Envelope, the DOCSIS Service Class Name Traffic Profile supports different Gate states as specified by the Envelope field, but each Envelope is defined by the same associated DOCSIS Service Class Name. This allows for having two-phase commit operations utilizing the DOCSIS Service Class Names, but each Envelope must be identical. Also note, that it is possible to change the DOCSIS Service Class Name associated with a Gate, but that such a change applies to all Envelopes associated with a given Gate.

6.4.2.7.3   Best Effort Service

The Best Effort object defines the Traffic Profile associated with a Gate through an upstream DOCSIS-specific parameterization scheme. The Best Effort object MUST conform to the following specification:

Length = 68, 128, 188		S-Num = 7	S-Type = 3
Envelope	Reserved	Reserved	Reserved

Authorized Envelope			
Traffic Priority	Reserved		
Request Transmission Policy			
Maximum Sustained Traffic Rate			
Maximum Traffic Burst			
Minimum Reserved Traffic Rate			
Assumed Minimum Reserved Traffic Rate Packet Size	Reserved		
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			
Maximum Buffer			
Data Rate Unit Setting	SF AQM Disable	Classic AQM Latency Target	AQM Algorithm
IAQM Max Threshold		IAQM Ramp Exponent	Reserved
Reserved Envelope (optional)			
Traffic Priority	Reserved		
Request Transmission Policy			
Maximum Sustained Traffic Rate			
Maximum Traffic Burst			
Minimum Reserved Traffic Rate			
Assumed Minimum Reserved Traffic Rate Packet Size	Reserved		
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			
Maximum Buffer			
Data Rate Unit Setting	SF AQM Disable	Classic AQM Latency Target	AQM Algorithm
IAQM Max Threshold		IAQM Ramp Exponent	Reserved
Committed Envelope (optional)			
Traffic Priority	Reserved		
Request Transmission Policy			
Maximum Sustained Traffic Rate			
Maximum Traffic Burst			
Minimum Reserved Traffic Rate			
Assumed Minimum Reserved Traffic Rate Packet Size	Reserved		
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			
Maximum Buffer			
Data Rate Unit Setting	SF AQM Disable	Classic AQM Latency Target	AQM Algorithm

IAQM Max Threshold	IAQM Ramp Exponent	Reserved

Traffic Priority is a 1-byte unsigned integer field specifying the relative priority assigned to the Service Flow in comparison with other flows. This field is fully defined in section C.2.2.5.1 of [DOCSIS MULPIv3.0]. A default Traffic Priority of 0 SHOULD be used if a specific Traffic Priority value is not required.

Request/Transmission Policy is a 4-byte bit field as defined in section C.2.2.6.3 of [DOCSIS MULPIv3.0]. A default Request/Transmission policy of 0 SHOULD be used if a specific Request/Transmission Policy value is not required.

Maximum Sustained Traffic Rate is a 4-byte unsigned integer field specifying the rate parameter, in bits/sec, for [DOCSIS MULPIv3.0]. A value of 0 indicates that no explicitly-enforced Maximum Sustained Rate is requested. A default Maximum Sustained Traffic Rate of 0 SHOULD be used if a specific Maximum Sustained Traffic Rate is not required.

Maximum Traffic Burst is a 4-byte unsigned integer field specifying the token bucket size, in bytes, for a token-bucket-based rate limit for this Service Flow. This field is fully defined in section C.2.2.5.3 of [DOCSIS MULPIv3.0]. A default Maximum Traffic Burst of 3044 bytes SHOULD be used if a specific Maximum Traffic Burst is not required. The value of this parameter has no effect unless a non-zero value has been provided for the Maximum Sustained Traffic Rate parameter.

Minimum Reserved Traffic Rate is a 4-byte unsigned integer field specifying the minimum rate, in bits/sec, reserved for this Service Flow. This field is fully defined in section C.2.2.5.4 of [DOCSIS MULPIv3.0]. A default Minimum Reserved Traffic Rate of 0 SHOULD be used if a specific Minimum Reserved Traffic Rate is not required.

Assumed Minimum Reserved Traffic Rate Packet Size is a 2-byte unsigned integer field specifying an assumed minimum packet size, in bytes, for which the Minimum Reserved Traffic Rate will be provided for this flow. This field is fully defined in section C.2.2.5.5 of [DOCSIS MULPIv3.0]. A default Assumed Minimum Reserved Traffic Rate Packet Size of 0 SHOULD be used if a specific Assumed Minimum Reserved Traffic Rate Packet size is not required. Upon receipt of a value of 0 the CMTS MUST utilize its implementation-specific default size for this parameter, not 0 bytes.

Maximum Concatenated Burst is a 2-byte unsigned integer specifying the maximum concatenated burst (in bytes) which a Service Flow is allowed. This field is fully defined in section C.2.2.6.1 of [DOCSIS MULPIv3.0]. A value of 0 means there is no limit. A default Maximum Concatenated Burst of 1522 bytes SHOULD be used if a specific Maximum Concatenated Burst is not required.

Upstream Peak Traffic Rate is a 4-byte unsigned integer specifying the Peak traffic rate (in bits per second) which a Service Flow is allowed. This field is fully defined in section C.2.2.5.10.1 of [DOCSIS MULPIv3.0].

Attribute Masks define a specific set of attributes associated with a DOCSIS 3.0 service flow. The CMTS MUST ignore the bonded bit in the Required and Forbidden Attribute Mask objects if the cable modem associated with the service flow is operating in pre-3.0 DOCSIS mode. The Required Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by requiring certain attributes. This field is fully defined in section C.2.2.3.6 of [DOCSIS MULPIv3.0]. The Forbidden Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by forbidding certain attributes. This field is fully defined in section C.2.2.3.7 of [DOCSIS MULPIv3.0]. The CMTS is free to assign the service flow to any channel that satisfies the traffic profile if no channel is available that satisfies the Required Attribute Mask and Forbidden Attribute Mask for the service flow. The Attribute Aggregation Rule Mask provides guidance to the CMTS as to how it might use the attribute masks of individual channels to construct a dynamic bonding group for this service flow. This field is fully described in section "Service Flow Attribute Aggregation Rule Mask" of [DOCSIS MULPIv3.0]. As described in that section a default Attribute Aggregation Rule Mask of 0 SHOULD be used if specific Attribute Aggregation Rules are not required.

The Buffer Control parameters limit the maximum queue depth of a Service Flow. The service flow buffer holds the packets that are enqueued for transmission for the service flow. The size of the service flow buffer sets the maximum queue depth, an upper limit on the amount of data that can be enqueued for transmission at any time by the service flow. By providing the ability to control per-service flow buffers, the below Buffer Control parameters provide a means of balancing throughput and latency in a standardized and configurable manner.

Minimum Buffer is a 4-byte unsigned integer parameter that defines a lower limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.3 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Minimum Buffer defaults to a value of 0, which indicates that there is no lower limit.

Target Buffer is a 4-byte unsigned integer parameter that defines a desired value for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.4 of [DOCSIS MULPIv3.0]. If this parameter is omitted or set to a value of 0, the device selects any buffer size within the range of the Minimum and Maximum Buffers, via a vendor specific algorithm.

Maximum Buffer is a 4-byte unsigned integer parameter that defines an upper limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.5 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Maximum Buffer defaults to a value of no limit.

The following fields are fully defined in section C.2.2 of [DOCSIS MULPIv3.1].

Data Rate Unit Setting: This is a 1-byte value that allows for the interpretation of Maximum Sustained Traffic Rate, Minimum Reserved Traffic Rate, Peak Traffic Rate and Guaranteed Grant Rate TLVs in units of bps, or kbps, or Mbps or Gbps.

SF AQM Disable: This is a 1-byte value that provides a means of disabling AQM on a particular service flow.

Classic AQM Latency Target: This is a 1 byte unsigned integer; this encoding provides the latency target (in milliseconds) to be used for the AQM algorithm for this Service Flow.

AQM Algorithm: This is a 1 byte value; this parameter defines the AQM algorithm to be used by the Service Flow.

Immediate AQM Max Threshold: This is a 2-byte unsigned integer; this parameter provides the maximum threshold in microseconds of the ramp function used by the Immediate AQM algorithm.

Immediate AQM Range Exponent of Ramp Function: This is a 1-byte unsigned integer; this parameter provides the range in nanoseconds of the ramp function used by the Immediate AQM algorithm.

6.4.2.7.4   Non-Real-Time Polling Service

The Non-Real Time Polling object defines the Traffic Profile associated with an upstream Gate through a DOCSIS-specific parameterization scheme. The Non-Real Time Polling object MUST conform to the following specification:

Length = 64, 120, 176		S-Num = 7	S-Type = 4
Envelope	Reserved	Reserved	Reserved
Authorized Envelope			
Traffic Priority	Reserved		
Request Transmission Policy			
Maximum Sustained Traffic Rate			
Maximum Traffic Burst			
Minimum Reserved Traffic Rate			
Assumed Minimum Reserved Traffic Rate Packet Size		Maximum Concatenated Burst	
Nominal Polling Interval			
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			
Maximum Buffer			
Reserved Envelope (optional)			
Traffic Priority	Reserved		
Request Transmission Policy			

Maximum Sustained Traffic Rate	
Maximum Traffic Burst	
Minimum Reserved Traffic Rate	
Assumed Minimum Reserved Traffic Rate Packet Size	Maximum Concatenated Burst
Nominal Polling Interval	
Upstream Peak Traffic Rate	
Required Attribute Mask	
Forbidden Attribute Mask	
Attribute Aggregation Rule Mask	
Minimum Buffer	
Target Buffer	
Maximum Buffer	
Committed Envelope (optional)	
Traffic Priority	Reserved
Request Transmission Policy	
Maximum Sustained Traffic Rate	
Maximum Traffic Burst	
Minimum Reserved Traffic Rate	
Assumed Minimum Reserved Traffic Rate Packet Size	Maximum Concatenated Burst
Nominal Polling Interval	
Upstream Peak Traffic Rate	
Required Attribute Mask	
Forbidden Attribute Mask	
Attribute Aggregation Rule Mask	
Minimum Buffer	
Target Buffer	
Maximum Buffer	

Traffic Priority is a 1-byte unsigned integer field specifying the relative priority assigned to the Service Flow in comparison with other flows. This field is fully defined in section C.2.2.5.1 of [DOCSIS MULPIv3.0]. A default Traffic Priority of 0 SHOULD be used if a specific Traffic Priority value is not required.

Request/Transmission Policy is a 4-byte bit field as defined in section C.2.2.6.3 of [DOCSIS MULPIv3.0]. Note: For this Service Flow Scheduling Type there is no default value for Request/Transmission Policy and all values (including 0) have meaning in DOCSIS.

Maximum Sustained Traffic Rate is a 4-byte unsigned integer field specifying the rate parameter, in bits/sec, for a token-bucket-based rate limit for this Service Flow. This field is fully defined in section C.2.2.5.2 of [DOCSIS MULPIv3.0]. A value of 0 indicates that no explicitly-enforced Maximum Sustained Rate is requested. A default Maximum Sustained Traffic Rate of 0 SHOULD be used if a specific Maximum Sustained Traffic Rate is not required.

Maximum Traffic Burst is a 4-byte unsigned integer field specifying the token bucket size, in bytes, for a token-bucket-based rate limit for this Service Flow. This field is fully defined in section C.2.2.5.3 of [DOCSIS MULPIv3.0]. A default Maximum Traffic Burst of 3044 bytes SHOULD be used if a specific Maximum Traffic Burst is not required. The value of this parameter has no effect unless a non-zero value has been provided for the Maximum Sustained Traffic Rate parameter.

Minimum Reserved Traffic Rate is a 4-byte unsigned integer field specifying the minimum rate, in bits/sec, reserved for this Service Flow. This field is fully defined in section C.2.2.5.4 of [DOCSIS MULPIv3.0]. A default Minimum Reserved Traffic Rate of 0 SHOULD be used if a specific Minimum Reserved Traffic Rate is not required.

Assumed Minimum Reserved Traffic Rate Packet Size is a 2-byte unsigned integer field specifying an assumed minimum packet size, in bytes, for which the Minimum Reserved Traffic Rate will be provided for this flow. This field is fully defined in section C.2.2.5.5 of [DOCSIS MULPIv3.0]. A default Assumed Minimum Reserved Traffic Rate Packet Size of 0 SHOULD be used if a specific Assumed Minimum Reserved Traffic Rate Packet size is not required. Upon receipt of a value of 0 the CMTS MUST utilize its implementation-specific default size for this parameter, not 0 bytes.

Maximum Concatenated Burst is a 2-byte unsigned integer specifying the maximum concatenated burst (in bytes) which a Service Flow is allowed. This field is fully defined in section C.2.2.6.1 of [DOCSIS MULPIv3.0]. A value of 0 means there is no limit. A default Maximum Concatenated Burst of 1522 bytes SHOULD be used if a specific Maximum Concatenated Burst is not required.

Nominal Polling Interval is a 4-byte unsigned integer field specifying the nominal interval (in units of microseconds) between successive unicast request opportunities for this Service Flow on the upstream channel. This field is fully defined in section C.2.2.6.4 of [DOCSIS MULPIv3.0]. A default Nominal Polling Interval of 0 SHOULD be used if a specific Nominal Polling Interval is not required. Upon receipt of a value of 0 the CMTS MUST utilize its implementation-specific default size for this parameter, not 0 microseconds.

Upstream Peak Traffic Rate is a 4-byte unsigned integer specifying the Peak traffic rate (in bits per second) which a Service Flow is allowed. This field is fully defined in section C.2.2.5.10.1 of [DOCSIS MULPIv3.0].

Attribute Masks define a specific set of attributes associated with a DOCSIS 3.0 service flow. The CMTS MUST ignore the bonded bit in the Required and Forbidden Attribute Mask objects if the cable modem associated with the service flow is operating in pre-3.0 DOCSIS mode. The Required Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by requiring certain attributes. This field is fully defined in section C.2.2.3.6 of [DOCSIS MULPIv3.0]. The Forbidden Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by forbidding certain attributes. This field is fully defined in section C.2.2.3.7 of [DOCSIS MULPIv3.0]. The CMTS is free to assign the service flow to any channel that satisfies the traffic profile if no channel is available that satisfies the Required Attribute Mask and Forbidden Attribute Mask for the service flow. The Attribute Aggregation Rule Mask provides guidance to the CMTS as to how it might use the attribute masks of individual channels to construct a dynamic bonding group for this service flow. This field is fully described in section "Service Flow Attribute Aggregation Rule Mask" of [DOCSIS MULPIv3.0]. As described in that section a default Attribute Aggregation Rule Mask of 0 SHOULD be used if specific Attribute Aggregation Rules are not required.

The Buffer Control parameters limit the maximum queue depth of a Service Flow. The service flow buffer holds the packets that are enqueued for transmission for the service flow. The size of the service flow buffer sets the maximum queue depth, an upper limit on the amount of data that can be enqueued for transmission at any time by the service flow. By providing the ability to control per-service flow buffers, the below Buffer Control parameters provide a means of balancing throughput and latency in a standardized and configurable manner.

Minimum Buffer is a 4-byte unsigned integer parameter that defines a lower limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.3 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Minimum Buffer defaults to a value of 0, which indicates that there is no lower limit.

Target Buffer is a 4-byte unsigned integer parameter that defines a desired value for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.4 of [DOCSIS MULPIv3.0]. If this parameter is omitted or set to a value of 0, the device selects any buffer size within the range of the Minimum and Maximum Buffers, via a vendor-specific algorithm.

Maximum Buffer is a 4-byte unsigned integer parameter that defines an upper limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.5 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Maximum Buffer defaults to a value of no limit.

6.4.2.7.5   Real-Time Polling Service

The Real-Time Polling object defines the Traffic Profile associated with an upstream Gate through a DOCSIS-specific parameterization scheme. The Real-Time Polling object MUST conform to the following specification:

Length = 64, 120, 176	S-Num = 7	S-Type = 5

Envelope	Reserved	Reserved	Reserved
Authorized Envelope			
Request Transmission Policy			
Maximum Sustained Traffic Rate			
Maximum Traffic Burst			
Minimum Reserved Traffic Rate			
Assumed Minimum Reserved Traffic Rate Packet Size		Maximum Concatenated Burst	
Nominal Polling Interval			
Tolerated Poll Jitter			
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			
Maximum Buffer			
Reserved Envelope (optional)			
Request Transmission Policy			
Maximum Sustained Traffic Rate			
Maximum Traffic Burst			
Minimum Reserved Traffic Rate			
Assumed Minimum Reserved Traffic Rate Packet Size		Maximum Concatenated Burst	
Nominal Polling Interval			
Tolerated Poll Jitter			
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			
Maximum Buffer			
Committed Envelope (optional)			
Request Transmission Policy			
Maximum Sustained Traffic Rate			
Maximum Traffic Burst			
Minimum Reserved Traffic Rate			
Assumed Minimum Reserved Traffic Rate Packet Size		Maximum Concatenated Burst	
Nominal Polling Interval			
Tolerated Poll Jitter			
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			
Maximum Buffer			

Request/Transmission Policy is a 4-byte bit field as defined in section C.2.2.6.3 of [DOCSIS MULPIv3.0]. Note: For this Service Flow Scheduling Type there is no default value for Request/Transmission Policy and all values (including 0) have meaning in DOCSIS.

Maximum Sustained Traffic Rate is a 4-byte unsigned integer field specifying the rate parameter, in bits/sec, for a token-bucket-based rate limit for this Service Flow. This field is fully defined in section C.2.2.5.2 of [DOCSIS MULPIv3.0]. A value of 0 indicates that no explicitly-enforced Maximum Sustained Rate is requested. A default Maximum Sustained Traffic Rate of 0 SHOULD be used if a specific Maximum Sustained Traffic Rate is not required.

Maximum Traffic Burst is a 4-byte unsigned integer field specifying the token bucket size, in bytes, for a token-bucket-based rate limit for this Service Flow. This field is fully defined in section C.2.2.5.3 of [DOCSIS MULPIv3.0]. A default Maximum Traffic Burst of 3044 bytes SHOULD be used if a specific Maximum Traffic Burst is not required. The value of this parameter has no effect unless a non-zero value has been provided for the Maximum Sustained Traffic Rate parameter.

Minimum Reserved Traffic Rate is a 4-byte unsigned integer field specifying the minimum rate, in bits/sec, reserved for this Service Flow. This field is fully defined in section C.2.2.5.4 of [DOCSIS MULPIv3.0]. A default Minimum Reserved Traffic Rate of 0 SHOULD be used if a specific Minimum Reserved Traffic Rate is not required.

Assumed Minimum Reserved Traffic Rate Packet Size is a 2-byte unsigned integer field specifying an assumed minimum packet size, in bytes, for which the Minimum Reserved Traffic Rate will be provided for this flow. This field is fully defined in section C.2.2.5.5 of [DOCSIS MULPIv3.0]. A default Assumed Minimum Reserved Traffic Rate Packet Size of 0 SHOULD be used if a specific Assumed Minimum Reserved Traffic Rate Packet size is not required. Upon receipt of a value of 0 the CMTS MUST utilize its implementation-specific default size for this parameter, not 0 bytes.

Maximum Concatenated Burst is a 2-byte unsigned integer specifying the maximum concatenated burst (in bytes) which a Service Flow is allowed. This field is fully defined in section C.2.2.6.1 of [DOCSIS MULPIv3.0]. A value of 0 means there is no limit. A default Maximum Concatenated Burst of 1522 bytes SHOULD be used if a specific Maximum Concatenated Burst is not required.

Nominal Polling Interval is a 4-byte unsigned integer field specifying the nominal interval (in units of microseconds) between successive unicast request opportunities for this Service Flow on the upstream channel. This field is fully defined in section C.2.2.6.4 of [DOCSIS MULPIv3.0]. For this Service Flow Scheduling Type there is no default value for Nominal Polling Interval.

Tolerated Polling Jitter is a 4-byte unsigned integer field specifying the maximum amount of time that the unicast request interval may be delayed from the nominal periodic schedule (measured in microseconds). This field is fully defined in section C.2.2.6.5 of [DOCSIS MULPIv3.0]. The minimum non-zero allowed value is 800 μs. If the CMTS receives a Gate-Set message with Tolerated Polling Jitter not equal to zero and less than 800 μs, the CMTS MUST reply with a Gate-Set-Err message with a PacketCable Error-Code of "Invalid Field Value in Object". A default Tolerated Polling Jitter of 0 SHOULD be used if a specific Tolerated Polling Jitter is not required. Upon receipt of a value of 0 the CMTS MUST utilize its implementation-specific default size for this parameter, not 0 microseconds. If the Tolerated Polling Jitter is set to 0 in a gate, the CMTS MUST return the value of 0 in any Gate-Info-Ack message for that gate.

Upstream Peak Traffic Rate is a 4-byte unsigned integer specifying the Peak traffic rate (in bits per second) which a Service Flow is allowed. This field is fully defined in section C.2.2.5.10.1 of [DOCSIS MULPIv3.0].

Attribute Masks define a specific set of attributes associated with a DOCSIS 3.0 service flow. The CMTS MUST ignore the bonded bit in the Required and Forbidden Attribute Mask objects if the cable modem associated with the service flow is operating in pre-3.0 DOCSIS mode. The Required Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by requiring certain attributes. This field is fully defined in section C.2.2.3.6 of [DOCSIS MULPIv3.0]. The Forbidden Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by forbidding certain attributes. This field is fully defined in section C.2.2.3.7 of [DOCSIS MULPIv3.0]. The CMTS is free to assign the service flow to any channel that satisfies the traffic profile if no channel is available that satisfies the Required Attribute Mask and Forbidden Attribute Mask for the service flow. The Attribute Aggregation Rule Mask provides guidance to the CMTS as to how it might use the attribute masks of individual channels to construct a dynamic bonding group for this service

flow. This field is fully described in section "Service Flow Attribute Aggregation Rule Mask" of [DOCSIS MULPIv3.0]. As described there a default Attribute Aggregation Rule Mask of 0 SHOULD be used if specific Attribute Aggregation Rules are not required.

The Buffer Control parameters limit the maximum queue depth of a Service Flow. The service flow buffer holds the packets that are enqueued for transmission for the service flow. The size of the service flow buffer sets the maximum queue depth, an upper limit on the amount of data that can be enqueued for transmission at any time by the service flow. By providing the ability to control per-service flow buffers, the below Buffer Control parameters provide a means of balancing throughput and latency in a standardized and configurable manner.

Minimum Buffer is a 4-byte unsigned integer parameter that defines a lower limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.3 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Minimum Buffer defaults to a value of 0, which indicates that there is no lower limit.

Target Buffer is a 4-byte unsigned integer parameter that defines a desired value for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.4 of [DOCSIS MULPIv3.0]. If this parameter is omitted or set to a value of 0, the device selects any buffer size within the range of the Minimum and Maximum Buffers, via a vendor-specific algorithm.

Maximum Buffer is a 4-byte unsigned integer parameter that defines an upper limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.5 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Maximum Buffer defaults to a value of no limit.

6.4.2.7.6   Unsolicited Grant Service

The Unsolicited Grant object defines the Traffic Profile associated with an upstream Gate through a DOCSIS-specific parameterization scheme. The Unsolicited Grant object MUST conform to the following specification:

Length =52, 96, 140		S-Num = 7	S-Type = 6
Envelope	Reserved	Reserved	Reserved
Authorized Envelope			
Request Transmission Policy			
Unsolicited Grant Size		Grants/Interval	Reserved
Nominal Grant Interval			
Tolerated Grant Jitter			
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			
Maximum Buffer			
Reserved Envelope (optional)			
Request Transmission Policy			
Unsolicited Grant Size		Grants/Interval	Reserved
Nominal Grant Interval			
Tolerated Grant Jitter			
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			

Maximum Buffer		
Committed Envelope (optional)		
Request Transmission Policy		
Unsolicited Grant Size	Grants/Interval	Reserved
Nominal Grant Interval		
Tolerated Grant Jitter		
Upstream Peak Traffic Rate		
Required Attribute Mask		
Forbidden Attribute Mask		
Attribute Aggregation Rule Mask		
Minimum Buffer		
Target Buffer		
Maximum Buffer		

Request/Transmission Policy is a 4-byte bit field as defined in section C.2.2.6.3 of [DOCSIS MULPIv3.0]. Note: For this Service Flow Scheduling Type there is no default value for Request/Transmission Policy and all values (including 0) have meaning in DOCSIS. Bit 9 in the Request/Transmission Policy enables/disables the use of segment headers. A segment header is 8 bytes in length. It MUST be accounted for in the Unsolicited Grant Size parameter when segment header usage is enabled. The CMTS MUST ignore Bit 9 in the Request/Transmission Policy if the cable modem associated with the service flow is operating in DOCSIS 1.1 or 2.0 mode. For more information on segment headers and their use please see section 6.3 of [DOCSIS MULPIv3.0].

Unsolicited Grant Size is a 2-byte unsigned integer field specifying the grant size (in bytes) as defined in section C.2.2.6.6 of [DOCSIS MULPIv3.0]. There is no default value of Unsolicited Grant Size.

Grants per Interval is a 1-byte unsigned integer field specifying the number of grants per Nominal Grant Interval as defined in section C.2.2.6.9 of [DOCSIS MULPIv3.0]. There is no default value of Grants per Interval, but a value of 1 is recommended.

Nominal Grant Interval is a 4-byte unsigned integer field specifying the nominal time between successive data grant opportunities for this Service Flow (in units of microseconds) as defined in section C.2.2.6.7 of [DOCSIS MULPIv3.0].

Tolerated Grant Jitter is a 4-byte unsigned integer field specifying the maximum amount of time that transmission opportunities may be delayed from the nominal periodic schedule (in units of microseconds) as defined in section C.2.2.6.8 of [DOCSIS MULPIv3.0]. The minimum allowed value is 800 μs. If the CMTS receives a Gate-Set message with Tolerated Grant Jitter less than 800 μs, the CMTS MUST reply with a Gate-Set-Err message with a PacketCable Error-Code of "Invalid Field Value in Object". There is no default value of Nominal Grant Interval. There is no default value for Tolerated Grant Jitter.

Upstream Peak Traffic Rate is a 4-byte unsigned integer specifying the Peak traffic rate (in bits per second) which a Service Flow is allowed. This field is fully defined in section C.2.2.5.10.1 of [DOCSIS MULPIv3.0].

Attribute Masks define a specific set of attributes associated with a DOCSIS 3.0 service flow. The CMTS MUST ignore the bonded bit in the Required and Forbidden Attribute Mask objects if the cable modem associated with the service flow is operating in pre-3.0 DOCSIS mode. The Required Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by requiring certain attributes. This field is fully defined in section C.2.2.3.6 of [DOCSIS MULPIv3.0]. The Forbidden Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by forbidding certain attributes. This field is fully defined in section C.2.2.3.7 of [DOCSIS MULPIv3.0]. The CMTS is free to assign the service flow to any channel that satisfies the traffic profile if no channel is available that satisfies the Required Attribute Mask and Forbidden Attribute Mask for the service flow. The Attribute Aggregation Rule Mask provides guidance to the CMTS as to how it might use the attribute masks of individual channels to construct a dynamic bonding group for this service flow. This field is fully described in section "Service Flow Attribute Aggregation Rule Mask" of [DOCSIS MULPIv3.0]. As described in that section, a default Attribute Aggregation Rule Mask of 0 SHOULD be used if specific Attribute Aggregation Rules are not required.

The Buffer Control parameters limit the maximum queue depth of a Service Flow. The service flow buffer holds the packets that are enqueued for transmission for the service flow. The size of the service flow buffer sets the maximum queue depth, an upper limit on the amount of data that can be enqueued for transmission at any time by the service flow. By providing the ability to control per-service flow buffers, the below Buffer Control parameters provide a means of balancing throughput and latency in a standardized and configurable manner.

Minimum Buffer is a 4-byte unsigned integer parameter that defines a lower limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.3 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Minimum Buffer defaults to a value of 0, which indicates that there is no lower limit.

Target Buffer is a 4-byte unsigned integer parameter that defines a desired value for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.4 of [DOCSIS MULPIv3.0]. If this parameter is omitted or set to a value of 0, the device selects any buffer size within the range of the Minimum and Maximum Buffers, via a vendor-specific algorithm.

Maximum Buffer is a 4 byte unsigned integer parameter that defines an upper limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.5 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Maximum Buffer defaults to a value of no limit.

6.4.2.7.7   Unsolicited Grant Service with Activity Detection

The Unsolicited Grant with Activity Detection object defines the Traffic Profile associated with an upstream Gate through a DOCSIS-specific parameterization scheme. The Unsolicited Grant with Activity Detection object MUST conform to the following specification:

Length = 60, 112, 164		S-Num = 7	S-Type = 7
Envelope	Reserved	Reserved	Reserved
Authorized Envelope			
Request Transmission Policy			
Unsolicited Grant Size		Grants/Interval	Reserved
Nominal Grant Interval			
Tolerated Grant Jitter			
Nominal Polling Interval			
Tolerated Poll Jitter			
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			
Maximum Buffer			
Reserved Envelope (optional)			
Request Transmission Policy			
Unsolicited Grant Size		Grants/Interval	Reserved
Nominal Grant Interval			
Tolerated Grant Jitter			
Nominal Polling Interval			
Tolerated Poll Jitter			
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			

Target Buffer		
Maximum Buffer		
Committed Envelope (optional)		
Request Transmission Policy		
Unsolicited Grant Size	Grants/Interval	Reserved
Nominal Grant Interval		
Tolerated Grant Jitter		
Nominal Polling Interval		
Tolerated Poll Jitter		
Upstream Peak Traffic Rate		
Required Attribute Mask		
Forbidden Attribute Mask		
Attribute Aggregation Rule Mask		
Minimum Buffer		
Target Buffer		
Maximum Buffer		

Request/Transmission Policy is a 4-byte bit field as defined in section C.2.2.6.3 of [DOCSIS MULPIv3.0]. Note: For this Service Flow Scheduling Type there is no default value for Request/Transmission Policy and all values (including 0) have meaning in DOCSIS. Bit 9 in the Request/Transmission Policy enables/disables the use of segment headers. A segment header is 8 bytes in length. It MUST be accounted for in the Unsolicited Grant Size parameter when segment header usage is enabled. The CMTS MUST ignore Bit 9 in the Request/Transmission Policy if the cable modem associated with the service flow is operating in DOCSIS 1.1 or 2.0 mode. For more information on segment headers and their use please see section 6.3 of [DOCSIS MULPIv3.0].

Unsolicited Grant Size is a 2-byte unsigned integer field specifying the grant size (in bytes) as defined in section C.2.2.6.6 of [DOCSIS MULPIv3.0] There is no default value of Unsolicited Grant Size.

Grants per Interval is a 1-byte unsigned integer field specifying the number of grants per Nominal Grant Interval as defined in section C.2.2.6.9 of [DOCSIS MULPIv3.0]. There is no default value of Grants per Interval, but a value of 1 is recommended.

Nominal Grant Interval is a 4-byte unsigned integer field specifying the nominal time between successive data grant opportunities for this Service Flow (in units of microseconds) as defined in section C.2.2.6.7 of [DOCSIS MULPIv3.0]. There is no default value of Nominal Grant Interval.

Tolerated Grant Jitter is a 4-byte unsigned integer field specifying the maximum amount of time that transmission opportunities may be delayed from the nominal periodic schedule (in units of microseconds) as defined in section C.2.2.6.8 of [DOCSIS MULPIv3.0]. The minimum allowed value is 800 μs. If the CMTS receives a Gate-Set message with Tolerated Grant Jitter less than 800 μs, the CMTS MUST reply with a Gate-Set-Err message with a PacketCable Error-Code of "Invalid Field Value in Object". There is no default value of Tolerated Grant Jitter.

Nominal Polling Interval is a 4-byte unsigned integer field specifying the nominal interval (in units of microseconds) between successive unicast request opportunities for this Service Flow on the upstream channel. This field is fully defined in section C.2.2.6.4 of [DOCSIS MULPIv3.0]. There is no default value of Nominal Polling Interval.

Tolerated Polling Jitter is a 4-byte unsigned integer field specifying the maximum amount of time that the unicast request interval may be delayed from the nominal periodic schedule (measured in microseconds). This field is fully defined in section C.2.2.6.5 of [DOCSIS MULPIv3.0]. The minimum non-zero allowed value is 800 μs. If the CMTS receives a Gate-Set message with Tolerated Polling Jitter not equal to zero and less than 800 μs, the CMTS MUST reply with a Gate-Set-Err message with a PacketCable Error-Code of "Invalid Field Value in Object". Upon receipt of a value of 0 the CMTS MUST utilize its implementation-specific default size for this parameter, not 0 microseconds.

Upstream Peak Traffic Rate is a 4-byte unsigned integer specifying the Peak traffic rate (in bits per second) which a Service Flow is allowed. This field is fully defined in section C.2.2.5.10.1 of [DOCSIS MULPIv3.0].

Attribute Masks define a specific set of attributes associated with a DOCSIS 3.0 service flow. The CMTS MUST ignore the bonded bit in the Required and Forbidden Attribute Mask objects if the cable modem associated with the service flow is operating in pre-3.0 DOCSIS mode. The Required Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by requiring certain attributes. This field is fully defined in section C.2.2.3.6 of [DOCSIS MULPIv3.0]. The Forbidden Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by forbidding certain attributes. This field is fully defined in section C.2.2.3.7 of [DOCSIS MULPIv3.0]. The CMTS is free to assign the service flow to any channel that satisfies the traffic profile if no channel is available that satisfies the Required Attribute Mask and Forbidden Attribute Mask for the service flow. The Attribute Aggregation Rule Mask provides guidance to the CMTS as to how it might use the attribute masks of individual channels to construct a dynamic bonding group for this service flow. This field is fully described in section "Service Flow Attribute Aggregation Rule Mask" of [DOCSIS MULPIv3.0]. As described in that section a default Attribute Aggregation Rule Mask of 0 SHOULD be used if specific Attribute Aggregation Rules are not required.

The Buffer Control parameters limit the maximum queue depth of a Service Flow. The service flow buffer holds the packets that are enqueued for transmission for the service flow. The size of the service flow buffer sets the maximum queue depth, an upper limit on the amount of data that can be enqueued for transmission at any time by the service flow. By providing the ability to control per-service flow buffers, the below Buffer Control parameters provide a means of balancing throughput and latency in a standardized and configurable manner.

Minimum Buffer is a 4-byte unsigned integer parameter that defines a lower limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.3 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Minimum Buffer defaults to a value of 0, which indicates that there is no lower limit.

Target Buffer is a 4-byte unsigned integer parameter that defines a desired value for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.4 of [DOCSIS MULPIv3.0]. If this parameter is omitted or set to a value of 0, the device selects any buffer size within the range of the Minimum and Maximum Buffers, via a vendor specific algorithm.

Maximum Buffer is a 4-byte unsigned integer parameter that defines an upper limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.5 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Maximum Buffer defaults to a value of no limit.

6.4.2.7.8   Downstream Service

The Downstream object defines the Traffic Profile associated with a Gate through a downstream DOCSIS-specific parameterization scheme. The Downstream object MUST conform to the following specification:

Length = 68, 128, 188		S-Num = 7	S-Type = 8
Envelope	Reserved	Reserved	Reserved
Authorized Envelope			
Traffic Priority	Downstream Resequencing	Reserved	
Maximum Sustained Traffic Rate			
Maximum Traffic Burst			
Minimum Reserved Traffic Rate			
Assumed Minimum Reserved Traffic Rate Packet Size		Reserved	
Maximum Downstream Latency			
Downstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			
Maximum Buffer			

Data Rate Unit Setting	SF AQM Disable		Classic AQM Latency Target	AQM Algorithm
IAQM Max Threshold			IAQM Ramp Exponent	Reserved
Reserved Envelope (optional)				
Traffic Priority		Downstream Resequencing	Reserved	
Maximum Sustained Traffic Rate				
Maximum Traffic Burst				
Minimum Reserved Traffic Rate				
Assumed Minimum Reserved Traffic Rate Packet Size				
Maximum Downstream Latency				
Downstream Peak Traffic Rate				
Required Attribute Mask				
Forbidden Attribute Mask				
Attribute Aggregation Rule Mask				
Minimum Buffer				
Target Buffer				
Maximum Buffer				
Data Rate Unit Setting	SF AQM Disable		Classic AQM Latency Target	AQM Algorithm
IAQM Max Threshold			IAQM Ramp Exponent	Reserved
Committed Envelope (optional)				
Traffic Priority		Downstream Resequencing	Reserved	
Maximum Sustained Traffic Rate				
Maximum Traffic Burst				
Minimum Reserved Traffic Rate				
Assumed Minimum Reserved Traffic Rate Packet Size			Reserved	
Maximum Downstream Latency				
Downstream Peak Traffic Rate				
Required Attribute Mask				
Forbidden Attribute Mask				
Attribute Aggregation Rule Mask				
Minimum Buffer				
Target Buffer				
Maximum Buffer				
Data Rate Unit Setting	SF AQM Disable	Classic AQM Latency Target	AQM Algorithm	
IAQM Max Threshold		IAQM Ramp Exponent	Reserved	

Traffic Priority is a 1-byte unsigned integer field specifying the relative priority assigned to the Service Flow in comparison with other flows. This field is fully defined in section C.2.2.5.1 of [DOCSIS MULPIv3.0]. A default Traffic Priority of 0 SHOULD be used if a specific Traffic Priority value is not required.

Downstream Resequencing is a 1-byte unsigned integer field specifying the use of sequence numbers in downstream DOCSIS 3.0 service flows. This field is fully defined in section C.2.2.7.3 of [DOCSIS MULPIv3.0]. The CMTS MUST honor the requested Downstream Resequencing operation for all Gate requests. It is possible that the CMTS may receive conflicting downstream resequencing direction by the AM for Multicast Gate requests (e.g., multiple Multicast Gate requests for the same Multicast destination but with different downstream resequencing operation). In such a case the CMTS MUST either honor the Multicast Gate request or reject it with error code 35 (Multicast Gate Downstream Resequencing mismatch). For a Multicast Gate, the CMTS MUST ignore the Downstream

Resequencing object if the cable modem associated with the service flow is operating in MDF disabled mode. The CMTS MUST ignore the Downstream Resequencing object if the cable modem associated with the service flow is operating in pre-3.0 DOCSIS mode. DOCSIS 3.0 introduced the concept of downstream channel bonding where the CMTS can simultaneously transmit on multiple channels. Downstream channels may not all have the same amount of latency such that two packets scheduled simultaneously by the CMTS may not arrive simultaneously at the cable modem. The CMTS can insert sequence numbers in each DOCSIS packet header to allow the cable modem to re-order out of sequence packets. The cable modem will hold higher numbered packets while waiting for lower numbered packets to arrive. The maximum wait time is 18ms. Applications that can tolerate lost packets or applications that cannot tolerate packet latency of up to 18ms can disable the use of sequence numbers by setting the Downstream Resequencing value to 1.

Maximum Sustained Traffic Rate is a 4-byte unsigned integer field specifying the rate parameter, in bits/sec, for a token-bucket-based rate limit for this Service Flow. This field is fully defined in section C.2.2.5.2 of [DOCSIS MULPIv3.0]. A value of 0 indicates that no explicitly-enforced Maximum Sustained Rate is requested. A default Maximum Sustained Traffic Rate of 0 SHOULD be used if a specific Maximum Sustained Traffic Rate is not required.

Maximum Traffic Burst is a 4-byte unsigned integer field specifying the token bucket size, in bytes, for a token-bucket-based rate limit for this Service Flow. This field is fully defined in section C.2.2.5.3 of [DOCSIS MULPIv3.0]. A default Maximum Traffic Burst of 3044 bytes SHOULD be used if a specific Maximum Traffic Burst is not required. The value of this parameter has no effect unless a non-zero value has been provided for the Maximum Sustained Traffic Rate parameter.

Minimum Reserved Traffic Rate is a 4-byte unsigned integer field specifying the minimum rate, in bits/sec, reserved for this Service Flow. This field is fully defined in section C.2.2.5.4 of [DOCSIS MULPIv3.0]. A default Minimum Reserved Traffic Rate of 0 SHOULD be used if a specific Minimum Reserved Traffic Rate is not required.

Assumed Minimum Reserved Traffic Rate Packet Size is a 2-byte unsigned integer field specifying an assumed minimum packet size, in bytes, for which the Minimum Reserved Traffic Rate will be provided for this flow. This field is fully defined in section C.2.2.5.5 of [DOCSIS MULPIv3.0]. A default Assumed Minimum Reserved Traffic Rate Packet Size of 0 SHOULD be used if a specific Assumed Minimum Reserved Traffic Rate Packet size is not required. Upon receipt of a value of 0 the CMTS MUST utilize its implementation-specific default size for this parameter, not 0 bytes.

Maximum Downstream Latency is a 4-byte unsigned integer field specifying the maximum latency between reception of a packet on the CMTS's NSI and the forwarding of the packet on its RF interface as defined in section C.2.2.7.1 of [DOCSIS MULPIv3.0]. A default Maximum Downstream Latency of 0 SHOULD be used if a specific Maximum Downstream Latency is not required. Upon receipt of a value of 0, the CMTS MUST NOT include this parameter in its DOCSIS signaling for this Service Flow.

Downstream Peak Traffic Rate is a 4-byte unsigned integer field, specifying the rate parameter P of a token-bucket-based peak rate limiter for packets of a downstream service flow. Configuring this peak rate parameter permits an operator to define a Maximum Burst value for the Downstream Maximum Sustained Rate much larger than a maximum packet size, but still limit the burst of packets consecutively transmitted for a service flow. The Downstream Peak Traffic Rate parameter is fully defined in section C.2.2.7.2 of [DOCSIS MULPIv3.0]. The CMTS MUST NOT include this parameter in its DOCSIS signaling for this service flow if a value of 0 is supplied, if the cable modem for which the Gate applies is not provisioned in DOCSIS 3.0 mode, or if the CMTS does not support the enforcement of this value.

Attribute Masks define a specific set of attributes associated with a DOCSIS 3.0 service flow. The CMTS MUST ignore the bonded bit in the Required and Forbidden Attribute Mask objects if the cable modem associated with the service flow is operating in pre-3.0 DOCSIS mode. The Required Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by requiring certain attributes. This field is fully defined in section C.2.2.3.6 of [DOCSIS MULPIv3.0]. The Forbidden Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by forbidding certain attributes. This field is fully defined in section C.2.2.3.7 of [DOCSIS MULPIv3.0]. The CMTS is free to assign the service flow to any channel that satisfies the traffic profile if no channel is available that satisfies the Required Attribute Mask and Forbidden Attribute Mask for the service flow. The Attribute Aggregation Rule Mask provides guidance to the CMTS as to how it might use the attribute masks of individual channels to construct a dynamic bonding group for this service

flow. This field is fully described in section "Service Flow Attribute Aggregation Rule Mask" of [DOCSIS MULPIv3.0]. As described in that section a default Attribute Aggregation Rule Mask of 0 SHOULD be used if specific Attribute Aggregation Rules are not required.

The Buffer Control parameters limit the maximum queue depth of a Service Flow. The service flow buffer holds the packets that are enqueued for transmission for the service flow. The size of the service flow buffer sets the maximum queue depth, an upper limit on the amount of data that can be enqueued for transmission at any time by the service flow. By providing the ability to control per-service flow buffers, the below Buffer Control parameters provide a means of balancing throughput and latency in a standardized and configurable manner.

Minimum Buffer is a 4-byte unsigned integer parameter that defines a lower limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.3 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Minimum Buffer defaults to a value of 0, which indicates that there is no lower limit.

Target Buffer is a 4-byte unsigned integer parameter that defines a desired value for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.4 of [DOCSIS MULPIv3.0]. If this parameter is omitted or set to a value of 0, the device selects any buffer size within the range of the Minimum and Maximum Buffers, via a vendor-specific algorithm.

Maximum Buffer is a 4-byte unsigned integer parameter that defines an upper limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.5 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Maximum Buffer defaults to a value of no limit.

The following fields are fully defined in section C.2.2 of [DOCSIS MULPIv3.1].

Data Rate Unit Setting: This is a 1-byte value that allows for the interpretation of Maximum Sustained Traffic Rate, Minimum Reserved Traffic Rate, Peak Traffic Rate and Guaranteed Grant Rate TLVs in units of bps, or kbps, or Mbps or Gbps.

SF AQM Disable : This is a 1-byte value that provides a means of disabling AQM on a particular service flow.

Classic AQM Latency Target: This is a 1 byte unsigned integer; this encoding provides the latency target (in milliseconds) to be used for the AQM algorithm for this Service Flow.

AQM Algorithm: This is a 1 byte value; this parameter defines the AQM algorithm to be used by the Service Flow.

Immediate AQM Max Threshold: This is a 2-byte unsigned integer; this parameter provides the maximum threshold in microseconds of the ramp function used by the Immediate AQM algorithm.

Immediate AQM Range Exponent of Ramp Function: This is a 1-byte unsigned integer; this parameter provides the range in nanoseconds of the ramp function used by the Immediate AQM algorithm.

6.4.2.7.9   Upstream Drop

The Upstream Drop object defines the Traffic Profile associated to a Gate with a null service flow. The Upstream Drop object MUST conform to the following specification:

Length = 8		S-Num = 7	S-Type = 9
Envelope = 7	Reserved	Reserved	Reserved

DOCSIS 3.0 [DOCSIS MULPIv3.0] section 7.5.1.2.2 details the CMTS and CM procedures when a Upstream Drop Traffic Profile is received from the AM via the PS.

An Upstream Drop object only applies to gates in the committed state; as a result, the AM MUST set the envelope of an Upstream Drop object to commit (7). If the CMTS receives an Upstream Drop object with an envelope other than commit (7), the CMTS MUST return PacketCable Error 12 (Incompatible Envelope) to the AM. When the CMTS realizes the Upstream Drop object is destined to a CM that does not support DOCSIS 3.0 or has UDC disabled, the CMTS MUST return a Gate-Set-Err with PacketCable Error 27 (Upstream Drop Unsupported) to the AM.

Since the Upstream Drop object represents a null service flow at the CM, the state timers managed by the CMTS are not applicable. As a result, the AM MUST disable (set value to 0) the state timers T1, T2, T3 and T4. If the CMTS

receives a value other than 0 for these timers, the CMTS MUST return PacketCable Error 17 (Invalid Field Value in Object) to the AM.

6.4.2.7.10   Proactive Grant Service

The PGS object defines the Traffic Profile associated with an upstream Gate through a DOCSIS-specific parameterization scheme.The PGS object MUST conform to the following specification:

Length = 76, 144, 212		S-Num = 7	S-Type = 10
Envelope	Reserved	Reserved	Reserved
Authorized Envelope			
Traffic Priority	Reserved		
Request Transmission Policy			
Maximum Sustained Traffic Rate			
Maximum Traffic Burst			
Minimum Reserved Traffic Rate			
Assumed Minimum Reserved Traffic Rate Packet Size	Reserved		
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			
Maximum Buffer			
Data Rate Unit Setting	SF AQM Disable	Classic AQM Latency Target	AQM Algorithm
IAQM Max Threshold		IAQM Ramp Exponent	PGS Activity Disable
Guaranteed Grant Interval (GGI)		Guaranteed Request Interval (GRI)	
Guaranteed Grant Rate (GGR)			
Reserved Envelope (optional)			
Traffic Priority	Reserved		
Request Transmission Policy			
Maximum Sustained Traffic Rate			
Maximum Traffic Burst			
Minimum Reserved Traffic Rate			
Assumed Minimum Reserved Traffic Rate Packet Size	Reserved		
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			
Maximum Buffer			
Data Rate Unit Setting	SF AQM Disable	Classic AQM Latency Target	AQM Algorithm
IAQM Max Threshold		IAQM Ramp Exponent	PGS Activity Disable
Guaranteed Grant Interval (GGI)		Guaranteed Request Interval (GRI)	
Guaranteed Grant Rate (GGR)			
Committed Envelope (optional)			
Traffic Priority	Reserved		

Request Transmission Policy			
Maximum Sustained Traffic Rate			
Maximum Traffic Burst			
Minimum Reserved Traffic Rate			
Assumed Minimum Reserved Traffic Rate Packet Size		Reserved	
Upstream Peak Traffic Rate			
Required Attribute Mask			
Forbidden Attribute Mask			
Attribute Aggregation Rule Mask			
Minimum Buffer			
Target Buffer			
Maximum Buffer			
Data Rate Unit Setting	SF AQM Disable	Classic AQM Latency Target	AQM Algorithm
IAQM Max Threshold		IAQM Ramp Exponent	PGS Activity Disable
Guaranteed Grant Interval (GGI)		Guaranteed Request Interval (GRI)	
Guaranteed Grant Rate (GGR)			

Traffic Priority is a 1-byte unsigned integer field specifying the relative priority assigned to the Service Flow in comparison with other flows. This field is fully defined in section C.2.2.5.1 of [DOCSIS MULPIv3.0]. A default Traffic Priority of 0 SHOULD be used if a specific Traffic Priority value is not required.

Request/Transmission Policy is a 4-byte bit field as defined in section C.2.2.6.3 of [DOCSIS MULPIv3.0]. Note: For this Service Flow Scheduling Type there is no default value for Request/Transmission Policy and all values (including 0) have meaning in DOCSIS. Bit 9 in the Request/Transmission Policy enables/disables the use of segment headers. A segment header is 8 bytes in length and MUST be accounted for in the Unsolicited Grant Size parameter when segment header usage is enabled. The CMTS MUST ignore Bit 9 in the Request/Transmission Policy if the cable modem associated with the service flow is operating in DOCSIS 1.1 or 2.0 mode. For more information on segment headers and their use please see section 6.3 of [DOCSIS MULPIv3.0].

Maximum Sustained Traffic Rate is a 4-byte unsigned integer field specifying the rate parameter, in bits/sec, for a token-bucket-based rate limit for this Service Flow. This field is fully defined in section C.2.2.5.2 of [DOCSIS MULPIv3.0]. A value of 0 indicates that no explicitly-enforced Maximum Sustained Rate is requested. A default Maximum Sustained Traffic Rate of 0 SHOULD be used if a specific Maximum Sustained Traffic Rate is not required.

Maximum Traffic Burst is a 4-byte unsigned integer field specifying the token bucket size, in bytes, for a token-bucket-based rate limit for this Service Flow. This field is fully defined in section C.2.2.5.3 of [DOCSIS MULPIv3.0]. A default Maximum Traffic Burst of 3044 bytes SHOULD be used if a specific Maximum Traffic Burst is not required. The value of this parameter has no effect unless a non-zero value has been provided for the Maximum Sustained Traffic Rate parameter.

Minimum Reserved Traffic Rate is a 4-byte unsigned integer field specifying the minimum rate, in bits/sec, reserved for this Service Flow. This field is fully defined in section C.2.2.5.4 of [DOCSIS MULPIv3.0]. A default Minimum Reserved Traffic Rate of 0 SHOULD be used if a specific Minimum Reserved Traffic Rate is not required.

Assumed Minimum Reserved Traffic Rate Packet Size is a 2-byte unsigned integer field specifying an assumed minimum packet size, in bytes, for which the Minimum Reserved Traffic Rate will be provided for this flow. This field is fully defined in section C.2.2.5.5 of [DOCSIS MULPIv3.0]. A default Assumed Minimum Reserved Traffic Rate Packet Size of 0 SHOULD be used if a specific Assumed Minimum Reserved Traffic Rate Packet size is not required. MMREQ15168 Upon receipt of a value of 0 the CMTS MUST utilize its implementation-specific default size for this parameter, not 0 bytes.

Upstream Peak Traffic Rate is a 4-byte unsigned integer specifying the Peak traffic rate (in bits per second) which a Service Flow is allowed. This field is fully defined in section C.2.2.5.10.1 of [DOCSIS MULPIv3.0].

Attribute Masks define a specific set of attributes associated with a DOCSIS 3.0 service flow. MMREQ15758 The CMTS MUST ignore the bonded bit in the Required and Forbidden Attribute Mask objects if the cable modem associated with the service flow is operating in pre-3.0 DOCSIS mode. The Required Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by requiring certain attributes. This field is fully defined in section C.2.2.3.6 of [DOCSIS MULPIv3.0]. The Forbidden Attribute Mask limits the set of channels and bonding groups to which the CMTS assigns the service flow by forbidding certain attributes. This field is fully defined in section C.2.2.3.7 of [DOCSIS MULPIv3.0]. The CMTS is free to assign the service flow to any channel that satisfies the traffic profile if no channel is available that satisfies the Required Attribute Mask and Forbidden Attribute Mask for the service flow. The Attribute Aggregation Rule Mask provides guidance to the CMTS as to how it might use the attribute masks of individual channels to construct a dynamic bonding group for this service flow. This field is fully described in section "Service Flow Attribute Aggregation Rule Mask" of [DOCSIS MULPIv3.0]. MMREQ15818 As described in that section, a default Attribute Aggregation Rule Mask of 0 SHOULD be used if specific Attribute Aggregation Rules are not required.

The Buffer Control parameters limit the maximum queue depth of a Service Flow. The service flow buffer holds the packets that are enqueued for transmission for the service flow. The size of the service flow buffer sets the maximum queue depth, an upper limit on the amount of data that can be enqueued for transmission at any time by the service flow. By providing the ability to control per-service flow buffers, the below Buffer Control parameters provide a means of balancing throughput and latency in a standardized and configurable manner.

Minimum Buffer is a 4-byte unsigned integer parameter that defines a lower limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.3 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Minimum Buffer defaults to a value of 0, which indicates that there is no lower limit.

Target Buffer is a 4-byte unsigned integer parameter that defines a desired value for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.4 of [DOCSIS MULPIv3.0]. If this parameter is omitted or set to a value of 0, the device selects any buffer size within the range of the Minimum and Maximum Buffers, via a vendor-specific algorithm.

Maximum Buffer is a 4 byte unsigned integer parameter that defines an upper limit for the size of the buffer that is to be provided for a service flow. This field is fully defined in section C.2.2.5.11.5 of [DOCSIS MULPIv3.0]. If this parameter is omitted, the Maximum Buffer defaults to a value of no limit.

The following fields are fully defined in section C.2.2 of [DOCSIS MULPIv3.1].

Data Rate Unit Setting: This is a 1-byte value that allows for the interpretation of Maximum Sustained Traffic Rate, Minimum Reserved Traffic Rate, Peak Traffic Rate and Guaranteed Grant Rate TLVs in units of bps, or kbps, or Mbps or Gbps.

SF AQM Disable : This is a 1-byte value that provides a means of disabling AQM on a particular service flow.

Classic AQM Latency Target: This is a 1 byte unsigned integer; this encoding provides the latency target (in milliseconds) to be used for the AQM algorithm for this Service Flow.

AQM Algorithm: This is a 1 byte value; this parameter defines the AQM algorithm to be used by the Service Flow

Immediate AQM Max Threshold: This is a 2-byte unsigned integer; this parameter provides the maximum threshold in microseconds of the ramp function used by the Immediate AQM algorithm.

Immediate AQM Range Exponent of Ramp Function: This is a 1-byte unsigned integer; this parameter provides the range in nanoseconds of the ramp function used by the Immediate AQM algorithm.

PGS Activity Disable is a 1 byte value that disables the activity detection function for an upstream service flow with the Proactive Grant Service scheduling type.

Guaranteed Grant Interval (GGI) is a 2-byte unsigned integer, this parameter specifies the maximum interval between successive data transmission opportunities, that is to be provided for a PGS Service Flow.

Guaranteed Grant Rate (GGR) is a 4-byte unsigned integer, this parameter specifies the minimum granting rate, in bits/sec, for an upstream PGS service flow. The valid range of this parameter is specific to the CMTS.

Guaranteed Request Interval (GRI) is a 2-byte unsigned integer, this parameter specifies the maximum interval (in units of microseconds) between successive request opportunities (including unicast request opportunities and piggyback request opportunities) for an upstream PGS service flow.

6.4.2.8   Event Generation Info

The Event Generation Info object contains all the information necessary to support the event messages as specified and required in [DOCSIS MULPIv3.0]. The IPv4 Event Generation Info object is used when the RKS Addresses are specified in IPv4 format, whereas the IPv6 Event Generation Info object is used when the RKS Addresses are specified in IPv6 format. Both Primary and Secondary RKS addresses MUST be specified using the same IP version.

The IPv4 Event Generation Info object MUST conform to the following specification:

Length = 44	S-Num = 8	S-Type = 1
Primary-Record-Keeping-Server-IP-Address (4-octets)		
Primary-Record-Keeping-Server-Port	Reserved	
Secondary-Record-Keeping-Server-IP-Address (4-octets)		
Secondary-Record-Keeping-Server-Port	Reserved	
Billing-Correlation-ID (24-bytes)		

Primary-Record-Keeping-Server-IP-Address is a 4-byte field which MUST contain the IPv4 address of the primary RKS to whom event records are to be sent.

Primary-Record-Keeping-Server-Port field is a 2-byte unsigned integer which MUST contain the port number on the primary RKS where event records are to be sent.

Secondary-Record-Keeping-Server-IP-Address is a 4-byte field which MUST contain the IPv4 address of the secondary RKS to whom records are to be sent if the primary RKS is unavailable.

Secondary-Record-Keeping-Server-Port is a 2-byte unsigned integer which MUST contain the port number on the secondary RKS where event records are to be sent.

Billing-Correlation-ID is a 24-byte field which MUST contain the identifier assigned by the AM or PS for all records related to this session. See [DOCSIS MULPIv3.0] for the definition and format of this attribute.

The IPv6 Event Generation Info object MUST conform to the following specification:

Length = 68	S-Num = 8	S-Type = 2
Primary-Record-Keeping-Server-IPv6-Address (16-octets)		
Primary-Record-Keeping-Server-Port	Reserved	
Secondary-Record-Keeping-Server-IPv6-Address (16-octets)		
Secondary-Record-Keeping-Server-Port	Reserved	
Billing-Correlation-ID (24-bytes)		

--	
--	
--	
--	

Primary-Record-Keeping-Server-IPv6-Address is a 16-byte field which MUST contain the IPv6 address of the primary RKS to whom event records are to be sent.

Primary-Record-Keeping-Server-Port field is a 2-byte unsigned integer which MUST contain the port number on the primary RKS where event records are to be sent.

Secondary-Record-Keeping-Server-IPv6-Address is a 16-byte field which MUST contain the IPv6 address of the secondary RKS to whom records are to be sent if the primary RKS is unavailable.

Secondary-Record-Keeping-Server-Port is a 2-byte unsigned integer which MUST contain the port number on the secondary RKS where event records are to be sent.

Billing-Correlation-ID is a 24-byte field which MUST contain the identifier assigned by the AM or PS for all records related to this session. See [DOCSIS MULPIv3.0] for the definition and format of this attribute.

6.4.2.9   Volume-Based Usage Limit

The Volume-Based Usage Limit object specifies the amount of data that can be transmitted over this Gate before meeting a volume threshold. This object is OPTIONAL in every Gate-Set message. It MUST be provided by the PEP in Gate-Info-Ack messages if it has been provided by the PDP in any Gate-Set message. It MUST NOT be provided in Gate-Info-Ack messages if it has not been provided by the PDP by any Gate-Set message. It MUST NOT be used in any other messages. The Volume-Based Usage Limit object MUST conform to the following specification:

Length = 12	S-Num = 9	S-Type = 1
Usage Limit		

Usage Limit is a 8-byte unsigned integer defined in units of kilobytes (1 kilobyte = 1024 bytes). A value of zero indicates no volume limit is imposed. The bytes counted towards the limit are from the byte after the DOCSIS MAC Header HCS to the end of the CRC for all packets transmitted on the Service Flow associated with this Gate.

6.4.2.10   Time-Based Usage Limit

The Time-Based Usage Limit object specifies the amount of time a Gate can remain committed before meeting a time limit threshold. The Time-Based Usage Limit object MUST conform to the following specification:

Length = 8	S-Num = 10	S-Type = 1
Time Limit		

Time Limit is a 4-byte unsigned integer defined in units of seconds. This is the limit on the amount of time a Gate can be in a committed state. This object is OPTIONAL in every Gate-Set message. If included in a Gate-Set this object MUST be stored by the CMTS. It MUST be provided by the PEP in Gate-Info-Ack messages if it has been provided by the PDP in any Gate-Set message. It MUST NOT be provided in Gate-Info-Ack messages if it has not been provided by the PDP by any Gate-Set message. While the Application Manager is ultimately responsible for Gates associated with a media session that has exceeded its Time-Based Usage Limit, the CMTS or Policy Server MAY use this object to police Application Manager enforcement of Time-Based Usage Limit policies established by the operator. The Application Manager or Policy Server MAY also query for this object as part of a failure recovery or other mechanism.

A value of zero indicates no time limit for the associated Gate.

6.4.2.11 Opaque Data

The Opaque Data object contains information that a Policy Server or Application Manager MAY store on a CMTS that remains opaque to the CMTS. The opaque data object is OPTIONAL in every Gate-Set message. It MUST NOT be used in any other messages issued by the PDP to the PEP. If the object has been defined by a successfully acknowledged Gate-Set message, the CMTS MUST store the Opaque Data locally. If Opaque Data has been defined, it MUST be returned by the CMTS in all appropriate acknowledgement, and Gate-Report-State messages. If the object has not been defined it MUST NOT be provided in acknowledgement, error, and Gate-Report-State messages. The Opaque Data object uses replace semantics, i.e., if the Opaque Data object has already been defined for a Gate, and a new Opaque Data object is received in a subsequent Gate-Set message for that Gate, the Opaque Data object in the subsequent Gate-Set message MUST replace the Opaque Data object currently being stored on the CMTS.

If the Opaque Data object is included in a Gate-Set message from the Application Manager to a Policy Server, the Policy Server MUST forward this object to the CMTS. The length of the Opaque Data is fixed at 8 bytes.

Length = 12	S-Num = 11	S-Type = 1
Opaque Data		

6.4.2.12 Gate Time Info

The Gate Time Info object contains the total amount of time the Gate has been in the Committed and Committed Recovery states. This counter MUST be stopped upon the Gate transitioning out of the Committed or Committed Recovery states to either the Reserved state or Authorized state. If the Gate subsequently transitions back to the Committed state, this counter MUST be restarted where it last stopped, i.e., when transitioning out of the Committed or Committed Recovery states. The Gate Time Info object MUST conform to the following specification:

Length = 8	S-Num = 12	S-Type = 1
Time Committed		

Time Committed is a 4-byte unsigned integer indicating the number of seconds this Gate has been in either the Committed state or the Committed-Recovery state. (Note: This is intended to be identical to docsQosServiceFlowTimeActive from the QOS MIB [IETF RFC 4546]).

6.4.2.13 Gate Usage Info

The Gate Usage Info object contains a counter indicating the number of kilobytes transmitted over this Gate. The Gate Usage Info object MUST conform to the following specification:

Length = 12	S-Num = 13	S-Type = 1
Octet Count		

Octet Count is a 8-byte unsigned integer which represents the number of bytes (counted from the DOCSIS MAC Header HCS to the end of the CRC) which have traversed the Service Flow associated with the Gate in units of 1024 bytes.

6.4.2.14 PacketCable Error

The PacketCable Error object contains information on the type of error that has occurred. The error is generated in response to a Gate Control command and is contained in Gate-Set-Err, Gate-Info-Err and Gate-Delete-Err messages. The PacketCable Error object MUST conform to the following specification:

Length = 8	S-Num = 14	S-Type = 1
Error-Code	Error-Subcode	

Error-Code is a 2-byte unsigned integer representing a specific error and MUST be one of the following:

Table 2 - Error Codes

Error Code	Description	Gate-Set-Err	Gate-Del-Err	Gate-Info-Err	Gate-Cmd-Err	PDP-Config-Err	Synch-Complete
1	Insufficient Resources	X					X
2	Unknown GateID	X	X	X			
6	Missing Required Object	X	X	X		X	X
7	Invalid Object	X	X	X		X	X
8	Volume Based Usage Limit Exceeded	X					
9	Time Based Usage Limit Exceeded	X					
10	Session Class Limit Exceeded	X					
11	Undefined Service Class Name	X					
12	Incompatible Envelope	X					
13	Invalid SubscriberID	X	X	X			X
14	Unauthorized AMID	X	X	X			X
15	Number of Classifiers Not Supported	X					
16	Policy Exception	X					
17	Invalid Field Value in Object	X					X
18	Transport Error	X	X	X			X
19	Unknown Gate Command				X		
20	DOCSIS 1.0 CM	X					
21	Number of SIDs exceeded in CM	X					
22	Number of SIDs exceeded in CMTS	X					
23	Unauthorized PSID	X	X	X		X	X
24	No State for PDP						X
25	Unsupported Synch Type						X
26	State Data Incomplete						X
27	Upstream Drop Unsupported	X					
28	Multicast Gate Error	X					
29	Multicast Volume Limit Unsupported	X					
30	Uncommitted Multicast Not Supported	X					
31	Multicast Gate Modification Not Supported	X					
32	Upstream Multicast Not Supported	X					
33	Multicast GateSpec incompatibility	X					
34	Multicast QoS Error	X					
35	Multicast Downstream Resequencing mismatch	X					
36	No match with provisioned service flows	X					
37	Ambiguous provisioned service flow Application ID	X					
38	Persistent Gate feature not supported or enabled	X					
39	Persistent Gate resources unavailable	X					
127	Other, Unspecified Error	X	X	X		X	X

Error-Subcode is a 2-byte unsigned integer field used to provide further information about the error. In the case of Error-Codes 6, 7 and 17, this 16-bit field MUST contain, as two 8-bit values the S-Num and S-Type of the object that is missing or in error. The order of the S-Num and S-Type values within the Error-Subcode MUST be the same as in the original message. In cases where multiple valid alternatives exist for the S-Type of a missing object, this portion of the Error-Subcode MUST be set to zero. In the case of Error-Code 15, the Error-Subcode field MUST contain the number of Classifiers supported per Gate. In the case of Error-Code 16, the Error-Subcode is a specific value assigned by the Policy Server. The values contained within the Error-Subcode field for Error-Code 16 are vendor specific and are not defined by this specification. In the case of Error-Code 19, the Error-Subcode is the Gate Command Type value that was contained in the original TransactionID object. For Error-Codes 36 and 37, the Error-Subcode is the low-order 16 bits of the ProvisionedSfApplicationId field from the Gate-Set.

Error-Codes 8, 9, 10, and 16 are generated as a result of a Policy Request failing to meet the requirements of a Policy Server's authorization. When the Application Manager issues a Gate-Set message with a volume or time-based limit to the Policy Server, the Policy Server MAY reject the request with Error-Code 8 or 9 based on policy rules installed on the Policy Server. For example, such a policy rule may state that if a volume limit request exceeds a maximum value, the Policy Server must reject the request. Error-Code 10 MAY be used if a policy is defined limiting parameters within the SessionClassID, without mapping them to alternate values. Error-Code 16 MAY be used to convey other policy exceptions in the Error-Subcode parameter. Error-Code 18 MAY be generated by a PEP if the gate command message is discarded because of congestion, buffer overflows, or link outages. Error-Code 19 MUST be generated by a PEP if the gate command message contains an unknown or invalid Gate Command Type value. Error-Code 1 (Insufficient Resources) MUST be used whenever a Gate is rejected because of issues related to the buffer control parameters.

6.4.2.15 Gate State

The information in the Gate State object reflects the current state of the Gate. The CMTS MUST include the Gate State object in any unsolicited messages that it sends to the Policy Server. The Policy Server may use this information to report state to the Application Manager, or for enforcing complex rules that might require state knowledge of the Gate.

Typically, the Policy Server is aware of state transitions since it usually provides the stimulus for these transitions to the CMTS, but in some cases the Gate may transition locally on the CMTS without the Policy Server's involvement. In these cases, the CMTS MUST report the state transition to the Policy Server via unsolicited Gate-Report-State messages. When issuing Gate-Report-State messages, the PEP MUST make sure the Solicited Flag in the COPS message header is cleared, and the Report Type in the header is set to Accounting. The Gate State object MUST conform with the following specification:

Length = 8		S-Num = 15	S-Type = 1
State		Reason	

State is a 2-byte unsigned integer field which MUST indicate one of the following states:

> 1 = Idle/Closed
> 2 = Authorized
> 3 = Reserved
> 4 = Committed
> 5 = Committed-Recovery

Reason is a 2-byte unsigned integer field which MUST indicate one of the following reasons for this update:

> 1 = Close initiated by CMTS due to reservation reassignment
> 2 = Close initiated by CMTS due to lack of DOCSIS MAC-layer responses
> 3 = Close initiated by CMTS due to timer T1 expiration
> 4 = Close initiated by CMTS due to timer T2 expiration
> 5 = Inactivity timer expired due to Service Flow inactivity (timer T3 expiration)
> 6 = Close initiated by CMTS due to lack of Reservation Maintenance
> 7 = Gate state unchanged, but volume limit reached
> 8 = Close initiated by CMTS due to timer T4 expiration
> 9 = Gate state unchanged, but timer T2 expiration caused reservation reduction

10 = Gate state unchanged, but time limit reached
11 = Close initiated by Policy Server or CMTS, volume limit reached
12 = Close initiated by Policy Server or CMTS, time limit reached
13 = Close initiated by CMTS, other
14 = Gate state unchanged, but SharedResourceID updated
15 = Close initiated by CMTS due to loss of shared resource
16 = Persistent Gate idle due to cable modem going offline
17 = Persistent Gate recreated by CMTS due to cable modem coming online
18 = Persistent Gate deleted from CMTS gate database
65535 = Other

6.4.2.16 Version Info

The Version Info object is used to enable Multimedia applications to adapt their interactions with other devices so that interoperability can be achieved between products supporting different protocol versions. Both the Major Version Number and the Minor Version Number are 2 byte unsigned integers. Both the PDP and the PEP must include this object as specified in Section 6.5.1.

Length = 8		S-Num = 16	S-Type = 1
Major Version Number		Minor Version Number	

6.4.2.17 PSID

PSID is a 4-byte unsigned integer value which uniquely identifies a Policy Server. The PSID number space is a subset of the AMID number space; therefore the PSID for a policy server MUST also be unique among all the AMID's that have been provisioned for Application Managers. The object MUST be formatted as follows:

Length = 8	S-Num = 17	S-Type = 1
PSID		

6.4.2.18 Synch Options

The Synch Options object is used to specify the details of how synchronization should be performed when a PDP issues a synchronization request. The object MUST be formatted as follows:

Length = 8		S-Num = 18	S-Type = 1
Reserved		Report Type	Synch Type

Report Type is a 1-byte unsigned integer used to indicate the type of data that should be sent back in synchronization reports and MUST be one of the following:

0 = Standard Report Data
1 = Complete Gate Data

Synch Type is a 1-byte unsigned integer used to indicate the type of synchronization that is being requested and MUST be one of the following:

0 = Full Synchronization
1 = Incremental Synchronization

6.4.2.19 Msg Receipt Key

The Msg Receipt Key is a 32-bit unsigned integer value, and is assigned by the PEP. This object MAY be included in any message sent from a PEP to a PDP and when it is present it is an indication that the PEP is requesting that the PDP confirms that it received the message. There are no constraints on how the PEP assigns this value, and for that reason the PDP should not make any assumptions about how it is used. The object MUST be formatted as follows:

Length = 8	S-Num = 19	S-Type = 1
Msg Receipt Key		

6.4.2.20 UserID

The UserID is a UTF-8 string value that identifies the user associated with a gate. This object is optional, and may be associated with a gate by an AM when the gate is originally created.

If the length (in bytes) of the string value is not a multiple of 4, then the value MUST be padded with null bytes until the length is a multiple of 4. Unlike most of the other objects in this specification where the Length field does not change, the Length field in the UserID object can vary. This field MUST be set to 4 more than the length of the padded value to account for the header fields.

Length = <Padded UserID length> + 4	S-Num = 21	S-Type = 1
UserID		
. . . as needed		

6.4.2.21 SharedResourceID

The Shared Resource Identifier is a 4-byte unsigned integer value assigned by the CMTS for Multicast Gate requests. There are no constraints, other than being locally unique within the scope of the CMTS, on how the CMTS assigns this value, and for that reason the PS and/or AM should not make any assumptions about how it is used, other than to identify when the same resource is used to fulfill multiple Gate requests.

The object MUST be formatted as follows:

Length = 8	S-Num = 22	S-Type = 1
SharedResourceID		

6.4.2.22 ProvisionedSfApplicationId

In order to reference and modify a provisioned service flow created during cable modem registration, the Application Manager MUST send a Gate-Set with the ProvisionedSfApplicationId set to a value that matches the Application Identifier (TLV 24/25.34) of a provisioned service flow. When a CMTS receives a Gate-Set with a ProvisionedSfApplicationId Information Element, the CMTS MUST attempt to match the value of ProvisionedSfApplicationId to the 32-bit Application Identifier of a provisioned DOCSIS service flow within the scope of the cable modem corresponding to the Subscriber ID. If a single matching service flow is found, then the CMTS MUST attempt to apply the gate to this service flow.If multiple matching service flows are found, then the CMTS MUST return a Gate-Set-Err message with the Error-Code 37 (Ambiguous provisioned service flow Application ID) and the Error-Subcode set to the low order 16 bits of ProvisionedSfApplicationId. If a matching service flow is not found, then the CMTS MUST return a Gate-Set-Err message with the Error-Code 36 (No match with provisioned service flows) and the Error-Subcode set to the low order 16 bits of ProvisionedSfApplicationId.

The object MUST be formatted as follows:

Length = 8	S-Num = 23	S-Type = 1
ProvisionedSfApplicationId		

6.4.3   Gate Control Messages

There are two separate profiles for Gate Control messages: one for messages exchanged between the Application Manager and the Policy Server, and one for messages between the Policy Server and the CMTS. While similar, these two profiles do exhibit some minor differences.

The following statements describe the PCMM message formats, covering both COPS and PCMM objects. These statements specify the content of messages, but do not imply a particular ordering of objects within each message. In particular, any ordering of PCMM objects MUST be accepted as valid (and may be generated), and the order of COPS objects MUST be as specified in [IETF RFC 2748]. Note that since PCMM objects only exist inside COPS objects, the distinction between these two sets is clear. This containment model also ensures that there are no issues with the relative order of COPS and PCMM objects.

6.4.3.1   Profile for Application Manager to Policy Server Interface

Messages that perform gate control between the Application Manager and Policy Server are defined and MUST be formatted as follows.

Note that messages from the Application Manager to Policy Server MUST be formatted as COPS Decision messages, and messages from Policy Server to Application Manager MUST be formatted as COPS Report-State messages.

<Client Open> = <COPS Common Header> <COPS PEPID> <ClientSI Info>

<ClientSI Info> = <COPS Client SI Header> <MM Version Info>

<Gate Control Command> = <COPS Common Header> <Client Handle> <Context>
                              <Decision Flags> <ClientSI Data>

<ClientSI Data> = <Gate-Set> | <Gate-Info> | <Gate-Delete>|
                  <PDP-Config> | <Synch-Request> | <Msg-Receipt>

<Gate Control Response> = <COPS Common Header> <Client Handle> <Report Type>
                              <ClientSI Object>

<ClientSI Object> = <Gate-Set-Ack> | <Gate-Set-Err> | <Gate-Info-Ack> | <Gate-Info-Err> |
                        <Gate-Delete-Ack> | <Gate-Delete-Err> | <Gate-Report-State>
                        <Gate-Cmd-Err> |
                        <PDP-Config-Ack> | <PDP-Config-Err> |
                        <Synch-Report> | <Synch-Complete>

<Gate-Set> = <Decision Header> <TransactionID> <AMID> <SubscriberID> [<GateID>]
                <GateSpec> <Traffic Profile> <classifier> [<classifier>...]
                [<Event Generation Info>]
                [<Volume-Based Usage Limit>] [<Time-Based Usage Limit>] [<Opaque Data>]
                [<UserID>] [<ProvisionedSfApplicationId>]

<Gate-Set-Ack> = <ClientSI Header> <TransactionID> <AMID> <SubscriberID> <GateID>
                [<Opaque Data>]
                [<Msg-Receipt-Key>]
                [<SharedResourceID>]

<Gate-Set-Err> = <ClientSI Header> <TransactionID> <AMID> <SubscriberID>
                <PacketCable Error> [<Opaque Data>] [<Msg-Receipt-Key>]

<Gate-Info> = <Decision Header> <TransactionID> <AMID> <SubscriberID><GateID>

<Gate-Info-Ack> = <ClientSI Header> <TransactionID> <AMID> <SubscriberID> <GateID>
                  [<Event Generation Info>] <GateSpec> <GateState> <classifier>
                  [<classifier...>] <Traffic Profile> [<UserID>]
                  <Gate Time Info><Gate Usage Info> [<Volume-Based Usage Limit>]
                  [<Time-Based Usage Limit>] [<Opaque Data>]
                  [<Msg-Receipt-Key>]
                  [<SharedResourceID>] [<ProvisionedSfApplicationId>]

        <Gate-Info-Err> = <ClientSI Header> <TransactionID> <AMID> <GateID> <PacketCable Err>

                      [<Opaque Data>] [<Msg-Receipt-Key>]

        <Gate-Delete> = <Decision Header> <TransactionID> <AMID> <SubscriberID> <GateID>

        <Gate-Delete-Ack> = <ClientSI Header> <TransactionID> <AMID> <GateID> [<Opaque Data>]

                      [<Msg-Receipt-Key>]

        <Gate-Delete-Err> = <ClientSI Header> <TransactionID> <AMID> <GateID>

                      <PacketCable Error> [<Opaque Data>] [<Msg-Receipt-Key>]

        <Gate-Report-State> = <ClientSI Header> <TransactionID> <AMID>

                      <SubscriberID> <GateID> <GateState>

                      <Gate Time Info> <Gate Usage Info> [<Opaque Data>]

                      [<Msg-Receipt-Key>] [<SharedResourceID>] [<ProvisionedSfApplicationId>]

        <Gate-Cmd-Err> = <ClientSI Header> <TransactionID> <AMID> <PacketCable Error>

        <PDP-Config> = <Decision Header> <TransactionID> <AMID> [<AMID>...]

        <PDP-Config-Ack> = <ClientSI Header> <TransactionID> [<Msg-Receipt-Key>]

        <PDP-Config-Err> = <ClientSI Header> <TransactionID> <PacketCable Error>

                      [<Msg-Receipt-Key>]

        <Synch-Request> = <Decision Header> <TransactionID> <AMID> [<SubscriberID>]

                      <Synch Options>

        <Synch-Report> = <ClientSI Header> <TransactionID> <AMID>

                      <SubscriberID> <GateID> <GateState>

                      <Gate Time Info> <Gate Usage Info> [<Opaque Data>]

                       [<GateSpec>] [<Traffic Profile>] [<classifier…>] [<Event Generation Info>]

                       [<Volume-Based Usage Limit>] [<Time-Based Usage Limit>]

                      [<Msg-Receipt-Key>] [<UserID>] [<SharedResourceID>]
                      [<ProvisionedSfApplicationId>]

        <Synch-Complete> = <ClientSI Header> <TransactionID> <AMID> [<PacketCable Error>]

                      [<Msg-Receipt-Key>]

        <Msg-Receipt> = <Decision Header> <TransactionID> <Msg-Receipt-Key>

6.4.3.2 *Profile for Policy Server to CMTS Interface*

Messages that perform Gate Control between the Policy Server and CMTS are defined and MUST be formatted as follows.

Note that messages from Policy Server to CMTS MUST be formatted as COPS Decision messages, and messages from CMTS to Policy Server MUST be formatted as COPS Report-State messages.

        <Client Open> = <COPS Common Header> <COPS PEPID> <ClientSI Info>

        <ClientSI Info> = <COPS Client SI Header> <MM Version Info>

        <Gate Control Command> = <COPS Common Header> <Client Handle> <Context>

                      <Decision Flags> <ClientSI Data>

        <ClientSI Data> = <Gate-Set> | <Gate-Info> | <Gate-Delete> |

<PDP-Config> | <Synch-Request> | <Msg-Receipt>

<Gate Control Response> = <COPS Common Header> <Client Handle> <Report Type>

<ClientSI Object>

<ClientSI Object> = <Gate-Set-Ack> | <Gate-Set-Err> | <Gate-Info-Ack> | <Gate-Info-Err> |

<Gate-Delete-Ack> | <Gate-Delete-Err> | < Gate-Report-State> |

<Gate-Cmd-Err> | <PDP-Config-Ack> | <PDP-Config-Err> |

<Synch-Report> | <Synch-Complete>

<Gate-Set> = <Decision Header> <TransactionID> <AMID> <SubscriberID> [<GateID>]

<GateSpec> <Traffic Profile> <classifier> [<classifier...>]

[<Event Generation Info>] [<Volume-Based Usage Limit>]

[<Time-Based Usage Limit>] [<Opaque Data>]

[<UserID>] [<ProvisionedSfApplicationId>]

<Gate-Set-Ack> = <ClientSI Header> <TransactionID> <AMID> <SubscriberID> <GateID>

[<Opaque Data>] [<Msg-Receipt-Key>] [<SharedResourceID>]

<Gate-Set-Err> = <ClientSI Header> <TransactionID> <AMID> <SubscriberID>

<PacketCable Error> [<Opaque Data>] [<Msg-Receipt-Key>]

<Gate-Info> = <Decision Header> <TransactionID> <AMID> <SubscriberID> <GateID>

[<PSID>]

<Gate-Info-Ack> = <ClientSI Header> <TransactionID> <AMID> <SubscriberID> <GateID>

[<Event Generation Info>] <Gate-Spec> <classifier> <classifier...>]

<Traffic Profile> <Gate Time Info> <Gate Usage Info>

[<Volume-Based Usage Limit>]

[<PSID>] [<Msg-Receipt-Key>] [<UserID>]

[<Time-Based Usage Limit>] [<Opaque Data>] <GateState>

[<SharedResourceID>] [<ProvisionedSfApplicationId>]

<Gate-Info-Err> = <ClientSI Header> <TransactionID> <AMID> <GateID> <PacketCable Err>

[<Opaque Data>] [<PSID>] [<Msg-Receipt-Key>]

<Gate-Delete> = <Decision Header> <TransactionID> <AMID> <SubscriberID> <GateID>

[<PSID>]

<Gate-Delete-Ack> = <ClientSI Header> <TransactionID> <AMID> <GateID> [<Opaque Data>]

[<PSID>] [<Msg-Receipt-Key>]

<Gate-Delete-Err> = <ClientSI Header> <TransactionID> <AMID> <GateID>

<PacketCable Error> [<Opaque Data>]

[<PSID>] [<Msg-Receipt-Key>]

<Gate-Report-State> = <ClientSI Header> <TransactionID> <AMID>

<SubscriberID> <GateID> <GateState>

<Gate Time Info> <Gate Usage Info> [<Opaque Data>]

[<Msg-Receipt-Key>]

[<SharedResourceID>] [<ProvisionedSfApplicationId>]

<Gate-Cmd-Err> = <ClientSI Header> <TransactionID> <AMID> <PacketCable Error>

<PDP-Config> = <Decision Header> <TransactionID> <PSID>

<PDP-Config-Ack> = <ClientSI Header> <TransactionID> [<Msg-Receipt-Key>]

<PDP-Config-Err> = <ClientSI Header> <TransactionID> <PacketCable Error> [<Msg-Receipt-Key>]

<Synch-Request> = <Decision Header> <TransactionID> [<AMID>] [<PSID>] [<SubscriberID>]

<Synch Options>

<Synch-Report> = <ClientSI Header> <TransactionID> <AMID> [<PSID>]

<SubscriberID> <GateID> <GateState>

<Gate Time Info> <Gate Usage Info> [<Opaque Data>]

[<GateSpec>] [<Traffic Profile>] [<Classifier…>] [<Event Generation Info>]

[<Volume-Based Usage Limit>] [<Time-Based Usage Limit>]

[<Msg-Receipt-Key>] [<UserID>] [<SharedResourceID>]
[<ProvisionedSfApplicationId>]

<Synch-Complete> = <ClientSI Header> <TransactionID> [<AMID>] [<PSID>]

[<PacketCable Error>] [<Msg-Receipt-Key>]

<Msg-Receipt> = <Decision Header> <TransactionID> <Msg-Receipt-Key>

There are six basic Gate Control command messages: Gate-Set, Gate-Info, Gate-Delete, PDP-Config, Synch-Request, and Msg-Receipt. These messages are embedded in the Client-Specific Decision Data in a COPS Decision message. For Gate Control command messages, the Context object (C-Num = 2, C-Type = 1) in the COPS Decision message MUST have the R-Type (Request Type Flag) value set to 0x08 (Configuration Request) and the M-Type set to zero. The Command-Code field in the mandatory Decision-Flags object (C-Num = 6, C-Type = 1) MUST be set to 1 (Install Configuration). Other values MUST cause the CMTS to generate a Report-State message indicating failure. The flags subfield MAY be set to any value and MUST be ignored by the PEP. The Gate Command Type field in the TransactionID object distinguishes the type of command being issued.

There are twelve Gate Control response messages: Gate-Set-Ack, Gate-Set-Err, Gate-Info-Ack, Gate-Info-Err, Gate-Delete-Ack, Gate-Delete-Err, Gate-Cmd-Err, PDP-Config-Ack, PDP-Config-Err, Synch-Report, Synch-Complete, and Gate-Report-State. The first eleven Gate Control response messages are solicited responses to Gate Control command messages. The twelfth, Gate-Report-State, is an unsolicited response to the PS from the CMTS to inform of a state change.

These messages are embedded in the Client-Specific Information Object in COPS Report-State messages. The Report-Type object (C-Num = 12, C-Type = 1) included in the COPS Report-State message for Gate Control responses MUST have the Report-Type field set to 1 (Success) or 2 (Failure) depending on the outcome of the Gate Control command. Report-State messages in response to a Gate Control command MUST have the solicited message flag bit set in the COPS header. The Gate Command Type field in the TransactionID object distinguishes the type of response being issued.

The CMTS generates the Gate-Report-State message when there is a state transition on the Gate that is not due to a Decision message, or when some policy limit has been reached. For the Gate-Report-State message, the Report-Type field MUST be set to 3 (Accounting), and the solicited flag field in the common header MUST be cleared.

If an object that is received in a Gate Control message contains an S-Num or S-Type that is invalid or unknown, that object MUST be ignored. An object is considered unknown if it has an S-Num/S-Type that is not defined in this specification. An object is considered invalid if it has a defined S-Num/S-Type and is present in a message for which that object type is not allowed. The presence of such an object within a Gate Control message MUST NOT be treated as an error provided that after such parameter is dropped, all required objects are present in the message.

6.5 Gate Control Protocol Operation

6.5.1 Initialization Sequence

When a PEP (Policy Server or CMTS) boots, it MUST listen for incoming COPS connections on the IANA-assigned TCP port number 3918. Any Application Manager or Policy Server (PDP) with a need to contact a PEP MUST initiate a TCP connection to the PEP on that port. It is expected that multiple Application Managers will establish COPS connections with multiple Policy Servers, and multiple Policy Servers will establish COPS connections with multiple CMTSs. When the TCP connection between the PEP and the PDP is established, the PEP MUST send information about itself to the PDP in the form of a Client-Open message. This message MUST include the Multimedia Version Info object, which will inform the PDP of the currently supported Multimedia protocol version used on the PEP.

Upon successful receipt of the Client-Open message, the PDP MUST send a Client-Accept message if the protocol version specified in the Version Info object is supported. This message MUST include the Keep-Alive-Timer object, which tells the PEP the maximum interval between Keep-Alive messages.

If the protocol version supplied by the PEP is not supported, the PDP MUST send a Client-Close messages with a COPS Error Object specifying error code 4 (Unable to process). After sending the Client-Close, the PDP MUST retain the TCP connection and security association with the PEP so that the PEP can reattempt the COPS initialization without reestablishing the TCP connection and security association. After receiving a Client-Close message from the PDP, which includes a COPS Error Object specifying error code 4, the PEP MAY reattempt initialization of the COPS connecting by sending another Client-Open message with another version number in the Version Info Object. This process MAY continue until the PEP has either received a Client-Accept message from the PDP, or has exhausted all available protocol versions. Once the PEP has tried all the protocol versions it supports, the PEP MUST send a Client-Open message with a Major Version Number of 0 and a Minor Version Number of 0 to indicate that it has unsuccessfully completed the version negotiation process. The PDP MUST then send a Client-Close message to the PEP to acknowledge that protocol negotiation has failed. On receipt of the Client-Close, the PEP MUST close the TCP connection. At this point, the PDP MAY periodically attempt to re-establish the connection.

Devices compliant with this specification MUST use a version of 6.0, i.e., a Version Info Object with a Major Version Number of 6 and a Minor Version Number of 0.

Upon successful receipt of the Client-Accept message, the PEP MUST send a Request message, including the Client-Handle and Context objects. The Context object (C-Num = 2, C-Type = 1) MUST have the R-Type (Request Type Flag) value set to 0x08 (Configuration Request) and M-Type set to zero. The Client-Handle object contains a value that MUST be chosen by the PEP. The only requirement imposed on this value is that a PEP MUST NOT use the same value for two different Requests on a single TCP connection. This completes the initialization sequence, which is visually depicted below.
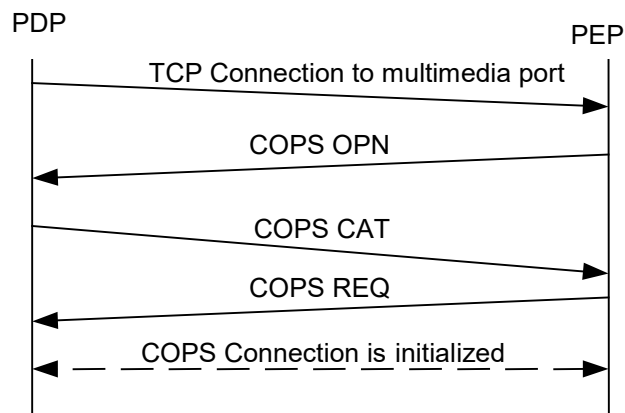
Figure 5 - COPS Connection Establishment

Periodically, the PEP MUST send a COPS Keep-Alive (KA) message to the PDP. Upon receipt of the COPS KA message, the PDP MUST echo a COPS KA message back to the PDP. This transaction is shown below and is fully documented in [DOCSIS MULPIv3.0]. The PEP MUST send a Keep-Alive message at least as often as specified in the Keep-Alive-Timer object returned in the Client-Accept message. The Keep-Alive message MUST be sent with Client-Type set to zero and the solicited flag cleared.
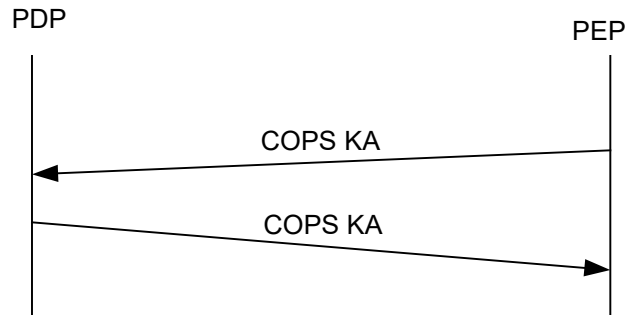
Figure 6 - COPS Keep-Alive Exchange

6.5.2   Operation Sequence

The protocol between the PDP and PEP is used for purposes of resource control and resource allocation policy. The Application Manager requests policy decisions from the Policy Server, and the Policy Server authorizes the requests and install them on the CMTS for enforcement through the use of Gates.

Messages that MAY be initiated by the Application Manager and Policy Server include Gate-Set, Gate-Info and Gate-Delete. The CMTS MAY initiate Gate-Report-State messages. The procedures for these messages are described in the following sections. All messages from the PDP to the PEP MUST be sent using Client-Specific objects within the Decision object of a COPS Decision message. Solicited responses from the PEP MUST be sent as a Report-State message with Client-Specific objects in the ClientSI object, and the solicited flag MUST be set. Gate-Report-State messages from the CMTS MUST be sent as unsolicited Report-State messages via Client-Specific objects in the ClientSI object.

The Decision messages and Report-State messages MUST contain the same Client-Handle as provided in the initial Request sent by the CMTS when the COPS connection was initiated.

Gate-Set initializes and modifies all the policy and traffic parameters for the Gate and establishes billing information. Gate-Set may also be used to control and update the state of a Gate on the CMTS.

Gate-Info is a mechanism by which the Policy Server may query all the current state and parameter settings of an existing Gate.

Gate-Delete allows a Policy Server to delete a specific Gate and any associated Service Flow.

Gate-Report-State allows the CMTS to inform the Policy Server that the Gate has transitioned into a new state. Gate-Report-State messages MUST be generated when the state transition happens asynchronously (i.e., not as a response to a Gate-Set message). Gate-Report-State messages MUST NOT be generated when the state transition happens synchronously.

Each Gate Control message that is sent by a PDP MUST include the TransactionID object. If this object is missing, the Gate Control message MUST be discarded and no response message be generated. If the TransactionID object contains an unknown or invalid (e.g., Gate-Delete-Err or Gate-Report-State) Gate-Command-Type value, an Gate-Cmd-Err error message MUST be generated by the PEP. Gate-Cmd-Err takes precedence over all other errors.

If the Gate Control message contains invalid or unknown objects, these objects MUST be ignored. If the Gate Control message contains missing mandatory objects, or other errors, then the error MUST be reported in the corresponding error message. No precedence of order of error checking is mandated.

In addition to the TransactionID, each Gate Control error message defines a set of mandatory elements. For each mandatory object that is missing in the original Gate Control message, a corresponding object with a null value MUST be generated for the response. The null value for AMID, SubscriberID, and GateID is defined to be zero.

The PEP MUST periodically send a Keep-Alive (KA) message to the PDP to facilitate the detection of TCP connection failures. The PDP MUST keep track of when KAs are received. If the PDP has not received a KA from the PEP in the time interval specified in [DOCSIS MULPIv3.0] and the PDP has not received an error indication from the TCP connection, then the PDP MUST tear down the TCP connection and attempt to re-establish the TCP connection.

The following rules are used to route Gate Control messages through the PacketCable Multimedia framework. In particular, provisions are provided for passing Gate Control messages forward (i.e., AM-to-PS-to-CMTS) and backward (i.e., CMTS-to-PS-to-AM) through a complex layered network in which multiple instances of each element are interacting with elements in the adjoining layer(s).

Upon receipt of a Gate Control message from an AM, a PS will apply any provisioned policy rules and determine whether to admit or reject the request. If the request is successfully admitted, the PS MUST route the message to the appropriate CMTS based on the SubscriberID included in the message. This SubscriberID-to-CMTS mapping MAY be performed dynamically based on a query to the OSS infrastructure or MAY reflect pre-provisioned routing information related to the range(s) of IP subnets that are associated with each CMTS.

If a Gate Control request is rejected by the PS, an error response MUST be returned to the issuing AM over the connection on which the original request was received. If a failure is detected on this connection between the time that a request is received and the response is delivered, the PS MUST discard the response.

Upon receiving a Gate Control message from a PS, a CMTS will execute the requested operation. If this operation is a successful Gate-Set operation, the CMTS MUST record the AMID and SubscriberID included in the message and maintain an association with the referenced Gate. This information must be used to ensure that only the AM which originally created the Gate is allowed to query or modify it. Any Gate Control messages that reference a Gate but which contain a AMID other than the one associated with the Gate MUST be rejected by the CMTS with error "Unauthorized AMID". If the PS that sent a Gate Control message had previously declared a PSID in a PDP-Config message then the CMTS MUST also maintain an association between that PSID and the referenced Gate. Gate-Report-State messages for a gate MUST be delivered to the PS element, identified through its PSID, that was last used to successfully perform a Gate-Set operation on that gate.

When a PS receives a Gate-Report-State message from a CMTS, the PS MUST forward this message to the AM associated with the AMID included in the message. In order to maintain a level of abstraction between non-adjacent layers and to hide information related to network topology from the AM layer, the PS MUST NOT include information directly identifying a particular CMTS to the AM layer.

6.5.3   Procedures for Validating Resource Envelopes

The set of service flow characteristics that are important for the purposes of providing enhanced Quality of Service is known as the envelope. A PacketCable Multimedia Gate contains up to three envelopes—one that indicates the Authorized resources, one that indicates the Reserved resources, and one that indicates the Committed resources for the service flow corresponding to the Gate.

The AM MUST ensure that the Committed envelope fits within the Reserved Envelope which fits within the Authorized envelope. For a Multicast Gate, the AM MUST set the Authorized, Reserved, and Committed envelopes identically when provided.

When a CMTS receives a Gate-Set message, it validates the relation between the Committed, Reserved, and Authorized envelopes of the Gate. The CMTS MUST confirm that the Committed envelope fits within the Reserved Envelope which fits within the Authorized envelope. For a Multicast Gate, the CMTS MUST also confirm that the Authorized envelope, Reserved envelope, and, where provided, Committed envelope are all identical. If the envelope relation is invalid, the CMTS MUST reply with a Gate-Set-Err message with a PacketCable Error-Code of "Incompatible Envelope".

For a Multicast Gate, the AM SHOULD provide the Authorized envelope, Reserved envelope, and Committed envelope in one Gate-Set message. If the Multicast Gate does not include a Committed envelope and the CMTS

does not support a multiphase committal of a Multicast gate, then the CMTS MUST reply with a Gate-Set-Err message with a PacketCable Error-Code of "Uncommitted Multicast Not Supported".

The CMTS MUST also perform admission control whenever a change (including an addition) of the reserved envelope is requested. Admission control is the process of assigning resources for the flow corresponding to the gate. If the resources cannot be assigned, the CMTS MUST reply with a Gate-Set-Err message with a PacketCable Error-Code of "Insufficient Resources".

6.5.3.1 *Flow Spec*

In Table 3, The second column indicates the operation that should be used to compare a parameter of A's envelope to a corresponding parameter in B's envelope. In other words, envelope A fits within envelope B if each of A's parameters meets the criteria specified in the table.

Table 3 - Envelope Comparison Rules

Parameter	A {OP} B
Token Bucket Rate [r]	≤
Token Bucket Size [b]	≤
Peak Data Rate [p]	≤
Minimum Policed Unit [m]	≥
Maximum Packet Size [M]	≤
Rate [R]	≤
Slack Term [S]	≥

6.5.3.2 *DOCSIS Service Class Name*

For traffic profiles in the form of a Service Class Name, the Service Class Name string MUST exactly match the preexisting Service Class Name on the CMTS. No envelope comparison is necessary as all three envelopes must share the same envelope parameters.

6.5.3.3 *DOCSIS Service Flow Parameters*

6.5.3.3.1 *Upstream Encodings*

In Table 4, the second column indicates the operation that should be used to compare a parameter of A's envelope to a corresponding parameter in B's envelope. In other words, envelope A fits within envelope B if each of A's parameters meets the criteria specified in the table.

Table 4 - Upstream Envelope Comparison

Parameter	A {OP} B
Traffic Priority (BE & NRTPS)	≤
Request Transmission Policy (all)	==
Maximum Sustained Traffic Rate (BE, NRTPS, RTPS)	≤
Maximum Traffic Burst (BE, NRTPS, RTPS)	≤
Minimum Reserved Traffic Rate (BE, NRTPS, RTPS)	≤
Assumed Minimum Reserved Traffic Rate Packet Size (BE, NRTPS, RTPS)	≥
Maximum Concatenated Burst (BE, NRTPS, RPTS)	≤
Nominal Polling Interval (NRTPS, RTPS, UGS/AD)	See interval description below
Tolerated Poll Jitter (RTPS, UGS/AD)	≥
Unsolicited Grant Size (UGS & UGS/AD)	≤
Grants per Interval (UGS & UGS/AD)	≤
Nominal Grant Interval (UGS & UGS/AD)	See interval description below

Parameter	A {OP} B
Tolerated Grant Jitter (UGS & UGS/AD)	≥
Minimum Buffer	≤
Target Buffer	Don't care
Maximum Buffer	≥
Data Rate Unit Setting	==
SF AQM Disable	Don't care
Classic AQM Latency Target	Don't care
AQM Algorithm	==
IAQM Max Threshold	Don't care
IAQM Ramp Exponent	Don't care
PGS Activity Disable	==
Guaranteed Grant Interval (GGI)	≥
Guaranteed Request Interval (GRI)	≥
Guaranteed Grant Rate (GGR)	≤

Intervals—A is a subset of B if the parameter in A is an integer multiple of the same parameter in B.

The Required, Forbidden, and Attribute Aggregation masks are ignored by the CMTS when comparing envelopes. Application Managers SHOULD set each of the Required, Forbidden, and Attribute Aggregation masks in a consistent way across the Authorized, Reserved, and Committed envelopes.

6.5.3.3.2   Downstream Encodings

In Table 5, the second column indicates the operation that should be used to compare a parameter of A's envelope to a corresponding parameter in B's envelope. In other words, envelope A fits within envelope B if each of A's parameters meets the criteria specified in the table.

Table 5 - Downstream Envelope Comparison

Parameter	A {OP} B
Traffic Priority	≤
Maximum Sustained Traffic Rate	≤
Maximum Traffic Burst	≤
Minimum Reserved Traffic Rate	≤
Assumed Minimum Reserved Traffic Rate Packet Size	≥
Maximum Downstream Latency	≥
Downstream Peak Traffic Rate	≤
Minimum Buffer	≤
Target Buffer	Don't care
Maximum Buffer	≥
Data Rate Unit Setting	==
SF AQM Disable	Don't care
Classic AQM Latency Target	Don't care
AQM Algorithm	==
IAQM Max Threshold	Don't care
IAQM Ramp Exponent	Don't care

The Required, Forbidden, and Attribute Aggregation masks are ignored by the CMTS when comparing envelopes. Application Managers SHOULD set each of the Required, Forbidden, and Attribute Aggregation masks in a consistent way across the Authorized, Reserved, and Committed envelopes.

6.5.3.4 *Upstream Drop*

Gate-Set commands with a traffic profile of type Upstream Drop do not contain envelope attributes. As a result, the CMTS need not perform resource envelope validations.

6.5.4 **Procedures for Authorizing Resources through a Gate**

The Gate-Set message MAY be sent by the PDP to the PEP to initialize or modify the operational parameters of a Gates. Figure 7 below provides an example of Gate-Set signaling. Note: As an example, the "Start Session" message can be used to indicate to the Client that the resources have been authorized.
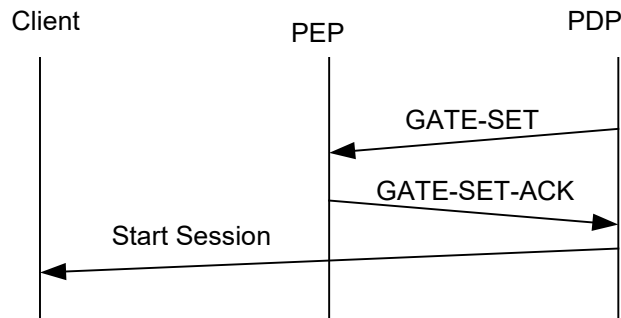
Figure 7 - Sample Signaling of Gate-Set

If a GateID object is present in the Gate-Set message, then the request is to modify an existing Gate. If the GateID object is missing from the Gate-Set message, then it is a request to allocate a new Gate. The Gate-Set message MUST contain exactly one GateSpec object, describing one upstream or downstream Gate.

The Gate-Set message also contains the SubscriberID. The CMTS MUST use this IP address (i.e., the SubscriberID) to determine the servicing CM and MUST use the MAC address of the CM for subsequent MAC-layer messaging. The CMTS MUST consider all SubscriberIDs which are routable via a CM as valid SubscriberIDs.

The PEP MUST respond to a Gate-Set message with either a Gate-Set-Ack, indicating success, or a Gate-Set-Err, indicating failure. The TransactionID in the response MUST match the TransactionID in the request. Errors in allocating or authorizing Gates MUST be reported by a Gate-Set-Err response. Refer to 6.4.2.14.

In Scenario 1, the Policy Server MAY specify the Authorized, Reserved and Committed Envelopes via a Traffic Profile sent in the Gate-Set message. It MAY simultaneously instruct the CMTS to authorize, reserve and commit resources.

Upon the receipt of a Gate-Set, the CMTS must first meet the requirements specified in Section 6.5.3 and then perform the requested actions. Upon successful completion of the actions requested in the Gate-Set (e.g., creation of a DOCSIS Service Flow) the CMTS MUST respond with a Gate-Set-Ack. The CMTS MUST NOT respond with a Gate-Set-Ack until it has completed sufficient steps to ensure that any subsequent requests to Admit or Commit the Gate will not fail due to a lack of resources.

A CMTS MAY perform complex authorization based not only on the requested QoS and the Gate's authorized FlowSpec, but also based on the SessionClassID specified in the GateSpec. The CMTS MAY have provisioned policies that define the amount of resources allocated exclusively to the particular Session Class, as well as 'borrow' and 'preemption' rules that apply to the use of the resources. The specifics of these types of policies and rules on the CMTS are out of scope for this specification.

Upon receipt of a Gate-Set-Ack or Gate-Set-Err from a CMTS, the Policy Server MUST forward the message to the Application Manager corresponding to the AMID in the Gate-Set-Ack. The Policy Server MUST NOT transmit a Gate-Set-Ack to an Application Manager prior to receiving a Gate-Set-Ack from the CMTS. If the Application

Manager requests a service that does not pass the Policy Server's policy checks, however, the Policy Server MUST NOT send the Gate-Set to the CMTS and MUST send a Gate-Set-Err to the Application Manager with the appropriate errors set.

6.5.4.1   Procedures for Authorizing Multicast Gates

In addition to the requirements specified in Section 6.5.4 above, the following requirements are added for Multicast Gate requests.

Upon receipt of a Gate-Set message without a GateID and with a classifier specifying a destination IP address of type Multicast, the CMTS MUST determine if this Multicast Gate request can be serviced by an existing SharedResourceID. If the Multicast Gate request is to be serviced by an existing resource with an assigned SharedResourceID, then the CMTS MUST include the identified SharedResourceID in the Gate-Set-Ack returned to the PS. If the Multicast Gate request will not be serviced by an existing resource with an assigned SharedResourceID, then the CMTS MUST determine whether it has resources available to service the Multicast Gate request. If the CMTS determines it can satisfy the request, it MUST create a new SharedResourceID and return it in the resulting Gate-Set-Ack.

DOCSIS 3.0 introduces a CMTS IP Multicast Join Authorization feature that allows operators to control which IP multicast sessions may be joined by multicast clients reached through a CM, by configuring rules on the CMTS. The CMTS MUST always authorize the IP multicast sessions signaled via PacketCable Multimedia Gate-Set messages to be joined by clients reached through a CM in spite of any authorization rules which may be configured locally on the CMTS.

The CMTS MUST use the QoS parameters signaled via Gate-Set message for the instantiation of the Group Service Flow associated with the multicast replication referenced in the Classifier specified in the Gate-Set message. The CMTS MUST ignore the CmtsGrpQosCfg entry associated with the classifier contained within the Multicast Gate request. If the Multicast Gate request signals a QoS envelope that is different than the QoS envelope associated with the SharedResourceID the CMTS has chosen to service the Multicast Gate request, the CMTS MAY reject the request with a Gate-Set-Err message with a PacketCable Error-Code of 34 (Multicast QoS Error).

If the CMTS cannot satisfy the Multicast Gate request (for any of the reasons herein, or due to local policy), it MUST return a Gate-Set-Err with an appropriate error code.

The CMTS MUST support Multicast Gates in the downstream direction. The CMTS MAY reject Multicast Gate requests which would result in an upstream Multicast Gate with error code 32 (Upstream Multicast Not Supported).

A Multicast Gate request will also contain a GateSpec object which defines the timers to be used for the gate state transitions as well as DSCP values and SessionClassID. It is possible that the CMTS may receive conflicting GateSpec parameters by the AM for Multicast Gate requests (e.g., multiple Multicast Gate requests for the same Multicast destination but with different timer values). In such a case the CMTS MUST either individually implement the GateSpec timer and DSCP values for each Multicast Gate request or reject Multicast Gate requests which do not match the DSCP and timer values associated with the underlying shared resource identified by the CMTS to service the Multicast Gate request. If the CMTS chooses to reject the Multicast Gate request due to incompatible GateSpec parameters, it MUST use error code 33 (Multicast GateSpec incompatibility).

6.5.5   Procedures for Querying a Gate

When a Policy Server or Application Manager wishes to query the current parameter settings of a Gate, it sends to the CMTS a Gate-Info message. The CMTS MUST respond to a Gate-Info message with either a Gate-Info-Ack, indicating success, or a Gate-Info-Err, indicating failure. A Gate-Info-Ack MUST contain all the current information on the Gate associated with the GateID in the Gate-Info message. If the Gate being queried has an existing Volume-Based and/or Time-Based Usage Limit, then the CMTS MUST include these objects in the Gate-Info-Ack. If the Gate being queried is a Multicast Gate, then the CMTS MUST include the current SharedResourceID in the Gate-Info-Ack. A PS or AM can utilize this information to recover Gate state information from the CMTS for policing, error recovery or other purpose. The TransactionID in the response MUST match the TransactionID in the request.

Errors in querying gates MUST be reported by a Gate-Info-Err response.

6.5.6   Procedures for Modifying a Gate

To modify the Traffic Profile associated with an existing Gate, an Application Manager MAY send a Gate-Set message with the GateID of the Gate to be modified and the new Traffic Profile. If the Gate-Set fails the Policy Server's checks, the Policy Server MUST send a Gate-Set-Err to the Application Manager and MUST NOT send a Gate-Set to the CMTS. However, if the Gate-Set passes the Policy Server's policy checks, the Policy Server MUST send the Gate-Set to the CMTS. The TransactionID in the Policy Server Gate-Set MUST match the TransactionID in the Application Manager Gate-Set.

Upon the receipt of a Gate-Set, the CMTS must first meet the requirements specified in Section 6.5.3 and then perform the requested actions. As with the creation of a new Gate, upon successful completion of the actions requested in the Gate-Set (e.g., modification of a DOCSIS service flow) the CMTS MUST respond with a Gate-Set-Ack. The CMTS MUST NOT respond with a Gate-Set-Ack until it has completed a sufficient number of steps to ensure that any subsequent requests to Admit or Commit the Gate will not fail due to lack of resources.

Upon receipt of a Gate-Set-Ack or Gate-Set-Err from the CMTS, the Policy Server MUST forward the response to the Application Manager.

To modify the Usage Limits associated with an existing Gate, an Application Manager MAY send a Gate-Set message with the GateID of the Gate to be modified. If the Traffic Profile in the Gate-Set is different from the Traffic Profile currently associated with the Gate, then the previous rules apply. In either case, if the Time-Based Usage Limit or Volume-Based Usage Limit of the flow is present, then the existing limits associated with this/these parameter(s) MUST be replaced with the new parameter(s). However, the absence of these parameters from a Gate-Set message indicates that even if the Traffic Profile for the Gate is being modified, the existing Time-Based or Volume-Based Usage Limit(s) of the Gate still apply. If these parameters are not present in a Gate-Set message, the existing limits MUST be maintained and their associated counters/timers MUST continue from their current value without resetting.

6.5.6.1   Procedures for Modification of Multicast Gates

If a Gate modification request for a Multicast Gate changes the resource requested by the Gate to be another shared or unshared resource, the CMTS MUST re-apply admission control logic for the newly requested resource, as it would during an initial Gate creation request, see Section 6.5.4.1. The CMTS MAY refuse to allow such transitions. When rejecting such gate modification requests, the CMTS MUST return a Gate-Set-Err with Error Code 31 (Multicast Gate Modification Not Supported). Note that a Classifier change (e.g., a Multicast Group Address Change) to the Gate may result in a change in shared resource usage for this Gate. The Gate-Set-Ack message MUST include the SharedResourceID for the Gate using the new Multicast Classifier.

If a Gate modification request for a Multicast Gate changes the source and/or destination IP address for a committed gate, and the CMTS supports such a change, the CMTS SHOULD treat the Gate-Set message as a "LeaveMulticastSession" for the old multicast session specified in the Gate and a "JoinMulticastSession" for the new multicast session specified in the Gate. For more information regarding handling of "JoinMulticastSession" and "LeaveMulticastSession" please refer to [DOCSIS MULPIv3.0].

6.5.7   Procedures for Supporting Usage Limits

The Application Manager, Policy Server and CMTS all have a role in enforcing Usage Limits. There are subtle differences between Time-Based and Volume-Based Limits so each of these is described separately.

The Volume-Based and Time-Based Usage Limits are relative to the Gate-Usage-Info and Gate-Time-Info values respectively. In order to support this, the Volume-Based and Time-Based Usage Limits are used as cumulative values. For example, if an Application Manager wants a Gate-Report-State generated when a Gate reaches 100 kilobytes of traffic, then it would send a Volume-Based Usage Limit of 100 kilobytes (if sent at the creation of the Gate, where the current value of Gate-Usage-Info is 0). If it then wanted a Gate-Report-State message generated when a Gate reaches 350 kilobytes, it would send a Volume-Based Usage Limit of 350 kilobytes, not 250 kilobytes. Along these lines, if an Application Manager wants to set a Volume-Based Usage Limit of 100 kilobytes after a Gate has already been created and has a non-zero value for Gate-Usage-Info, it should send a Gate-Info message to obtain the current value of Gate-Usage-Info add 100 kilobytes to that value, and use this new value in the Volume-Based Usage Limit in a subsequent Gate-Set.

Given the CMTS is the only device that tracks both Gate-Usage-Info and Gate-Time-Info, it is required to generate a Gate-Report-State notifying the AM when a Volume-Based or Time-Based Usage Limit is set less than or equal to the current Gate-Usage-Info or Gate-Time-Info value respectively. If the Volume-Based Usage Limit is set less than the current Gate-Usage-Info value, the CMTS MUST send a Gate-Report-State message with a Gate-State reason code of 7 (Gate state unchanged, but volume limit reached). If the Time-Based Usage Limit is set less than the current Gate-Time-Info value, the CMTS MUST send a Gate-Report-State message with a Gate-State reason code of 10 (Gate state unchanged, but time limit reached). This is the only case where the CMTS would send reason code 10 since the CMTS is not responsible for tracking Time Based Usage Limits, see Section 6.5.7.2. Using zero (0) for a Usage Limit is an exception to this, as it's used to disable the Usage Limit. In addition, the Gate-Usage-Info object within a Gate-Report-State message is not applicable to gates created with a traffic profile of Upstream Drop. The CMTS MUST return a value of zero (0) bytes.

Depending on the CMTS implementation, volume usage tracking and volume-based usage limits for individual Multicast Gates may be unsupported. If the CMTS does not support tracking volume usage per Multicast Gate, it MUST reject any Multicast Gate-Set message including a Volume Based Usage Limit. In such a case, the CMTS MUST specify error code 29 (Multicast Volume Limit Unsupported), in the resulting Gate-Set-Error message.

6.5.7.1   Procedures For Supporting Volume-Based Usage Limits

Since the CMTS is the only trusted PacketCable Multimedia device in the packet path, it is the only device capable of accurately tracking the usage of individual Gates. Thus, the CMTS MUST track the usage of all Gates regardless of whether or not they have an associated Volume-Based Usage Limit. The CMTS MAY track the usage of individual Multicast Gates. The CMTS MUST report the amount of data transferred via a Unicast Gate in all Gate-Info-Ack and all Gate-Report-State messages. If the CMTS does not support tracking volume usage on individual Multicast Gates, it MUST use a zero value for the Gate Usage Info object in all Gate-Info-Ack and Gate-Report-State messages, and also not include a Volume Based Usage Limit object.

If the Gate has an associated Volume-Based Usage Limit when the amount of data that has traversed the Gate equals the Volume-Based Usage Limit, the CMTS MUST send a Gate-Report-State message with the Solicited bit set to 0. The Gate-Report-State message MUST include a Gate State object with the Reason set to 7 (Gate state unchanged, but volume limit reached). Upon receipt of a Gate-Report-State message, the behavior of a PDP depends upon its role; a Policy Server MUST either forward the Gate-Report-State message to the Application Manager, or handle the report itself. The Policy Server SHOULD only handle the report if it caused the report to be generated by modifying the original gate set. In other words, the Policy Server's actions should be transparent to the Application Manager. The Application Manager MUST handle received reports. A PDF handles a Gate-Report-State message with the Reason set to 7 by performing one of the following actions:

- Send a Gate-Set message with a new Volume-Based Usage Limit object.

- Send a Gate-Set message with a Volume-Based Usage Limit set to 0 to explicitly disable the feature, or do nothing to implicitly disable the feature.

- Close the Gate by issuing a Gate-Delete command.

6.5.7.2   Procedures For Supporting Time-Based Usage Limits

While it is a desirable design goal to keep the Volume-Based and Time-Based Usage Limit procedures as similar as possible, the number of CMTS interrupts required to support enforcement of Time-Based Usage Limits by the CMTS makes this impractical. Thus, the Application Manager MUST enforce the Time-Based Usage Limit of the Gate. Upon receiving the Gate-Set-Ack for a Gate with a Time-Based Usage Limit, the AM MUST start an application timer. When the application timer is equal to the Time-Based Usage Limit, the Application Manager MUST respond by performing one of the following actions:

- Send a Gate-Set message with a new Time-Based Usage Limit object.

- Send a Gate-Set message with a Time-Based Usage Limit set to 0 to explicitly disable the feature, or do nothing to implicitly disable the feature.

- Close the Gate by issuing a Gate-Delete command.

Note: In some ways it makes more sense for the Application Manager to enforce Usage Limits, since the Time-Based Usage Limit and Volume-Based Usage Limit are a reflection of the service being offered and are the responsibility of the Service Control Domain. It is really the Volume-Based Usage Limit procedure which is unusual, but the CMTS is the only device that can accurately enforce this limit.

6.5.7.3   Resource & Error Recovery

While it is required that the Application Manager perform one of several actions when a Gate's Usage Limit has been reached, there is always the possibility that the Application Manager will not respond appropriately. In this case, the RKS will still be recording the usage of this Gate so this activity will still be billable, but in some instances it may be useful to recover the resources that are being used 'illegally' by the Application Manager. A Policy Server MAY glean the fact that the Volume-Based or Time-Based Usage Limit of a Gate has been exceeded based on the messages it is proxying between the AM and CMTS. Using the 'gleaning' technique implies that the Policy Server is stateful, but a Policy Server that is not stateful can still recover resources via a second technique described below.

Alternatively, a Policy Server MAY occasionally query the CMTS with a Gate-Info message. The response will contain any associated Volume-Based Usage Limit and Gate Usage Info (or Time-Based Usage Limit and Gate Time Info). The Policy Server can then compare these values. Regardless of how a Policy Server gains the knowledge that a Gate is over-limit, it MAY issue a Gate-Delete for over-limit Gates. Upon receipt of the Gate-Delete-Ack from the CMTS, the Policy Server MUST send a Gate-Report-State with the State set to 1 (Idle/Closed) and Reason set to 11 (Close initiated by Policy Server or CMTS, volume limit reached) or 12 (Close initiated by Policy Server or CMTS, time limit reached) to the Application Manager. If a Gate-Delete-Err is received from the CMTS, the Policy Server MUST NOT send anything to the Application Manager because the Gate state is not changed.

Similarly, while not required to recover resources from over-limit Gates, a CMTS MAY perform the same comparisons itself and MAY delete over-limit Gates. Additional requirements for this scenario are described in Section 6.5.8.

Given the fact that the Policy Server and the CMTS MAY delete over-limit Gates, it is recommended that the Application Manager, if it no longer wants to monitor a Volume-Based and/or Time-Based Usage Limit, send a Volume-Based and/or Time-Based Usage Limit of 0. This will explicitly disable the Volume-Based and/or Time-Based Usage Limit, and will indirectly inform the Policy Server and CMTS that the over-limit Gate has been acted upon.

6.5.7.4   Tracking Time-Based and Volume-Based Usage Limits

PacketCable Multimedia Gates can be committed and uncommitted multiple times (to support, for example, a 'pause' function on a game or streaming media). Since a subscriber cannot transmit/receive data while the Gate is not a committed state, these periods should not be counted against them. For Volume-Based Limits this requirement has no effect—there are no packets that might be over counted since no packets can be sent if a Gate is not committed. However, for Time-Based Usage Limits the CMTS MUST stop its Gate Time Info timer when the Gate is not in either the Committed State or the Committed-Recovery State. If the Gate is re-Committed, the Gate Time Info timer MUST be restarted from its stopped count.

Note: The Application Manager is required to maintain a timer independent of the CMTS timer to enforce the Time-Based Usage Limit. Since this timer is separated from the CMTS itself, messaging delays could cause discrepancies between these two timers. For applications that require a high-degree of time accuracy, the AM MAY query the CMTS for its Gate Time Info object after it moves a Gate into or out of a committed state.

6.5.8   Procedures for Deleting a Gate

Normally, when a Multimedia session ends, the Application Manager tells the Policy Server that the session has ended, and the Policy Server in turn instructs the CMTS to remove the Gate via a Gate-Delete message. The CMTS MUST respond to a Gate-Delete message with a Gate-Delete-Ack, indicating success, or a Gate-Delete-Err, indicating failure. The TransactionID in the response MUST match the TransactionID in the request.

Errors in deleting Gates MUST be reported by a Gate-Delete-Err response.

At the CMTS, if timer T1, T2 (only when in the Reserve state), or T4 expires, the Gate MUST be deleted. When a CMTS deletes a Gate without being solicited by the Policy Server, the CMTS MUST send a Gate-Report-State message (with the Solicited bit set to 0) to the Policy Server indicating that the Gate was deleted. If the T2 timer expires while in the Reserved state, the CMTS MUST delete the DOCSIS flow through DOCSIS mechanisms (i.e., a DSD message) and issue a Gate-Report-State message (with the Solicited bit set to 0) to the PS informing it of this state transition. Note, if the T2 timer expires while in the Committed or Committed-Recovery state then the CMTS must send a DSC as defined in DOCSIS to free the reserved resources that are in excess of the active resources, issue a Gate-Report-State message to the PS informing it of the reduction in reservation resources, and remain in the same state. Upon receipt of a Gate-Report-State message, the Policy Server MUST forward it to the Application Manager.

At the discretion of the Policy Server, for example, as described in section 6.5.7.3 for resource and error recovery, the Policy Server may send a Gate-Delete message to the CMTS to delete a gate.

6.5.8.1   Procedures for Deleting Multicast Gates

When a Multicast Gate is deleted, the CMTS MUST delete all information and state associated with the deleted gate and respond with a Gate-Delete-Ack. If the assigned SharedResourceID is still being used by other Multicast Gates, the CMTS MUST preserve the SharedResourceID and the resource it references. If the deleted Multicast Gate is the last Gate associated with the assigned SharedResourceID, the CMTS MAY delete all information and state associated with the SharedResourceID and the resource it references.

When a committed Multicast Gate is deleted, the CMTS SHOULD treat the Gate-Delete message as a "LeaveMulticastSession" for the multicast session specified in the Gate. For more information regarding handling of a "LeaveMulticastSession" refer to [DOCSIS MULPIv3.0].

6.5.9   Procedure for Committing a Gate

In Scenario 1, the Policy Server is responsible for committing a Gate through a Traffic Profile containing a Committed Envelope. The CMTS commits the Gate and activates the DOCSIS Service Flow using the parameters passed down to it by the Policy Server.

6.5.9.1   Procedure for Committing a Multicast Gate

When a Multicast Gate is committed, the CMTS SHOULD treat the commit operation as a "JoinMulticastSession" for the multicast session specified in the Gate. For more information regarding handling of a "JoinMulticastSession" please refer to [DOCSIS MULPIv3.0].

6.5.10   Termination Sequence

When the PEP is shutting down its TCP connection to the PDP, it MAY first send a Delete-Request-State (DRQ) message (including the Handle object used in the initial Request message). If the PEP chooses to send the DRQ message, the PEP MUST use COPS reason code 4 (Tear). The PEP MAY follow that with a Client-Close message. The PDP in response MUST automatically delete any state associated with the PEP when the TCP connection is terminated. When the PDP is going to shutdown, it SHOULD send a COPS Client-Close message to the PEP. In the COPS Client-Close message, the PDP SHOULD NOT send the PDP redirect address object PDPRedirAddr. If the PEP receives a COPS Client-Close message from the PDP with a PDPRedirAddr object, the PEP MUST ignore the PDPRedirAddr while processing the COPS Client-Close message.

The PS and CMTS MUST NOT remove gates as a result of a failed COPS connection.

6.5.11   Procedures for Multiple Grants Per Interval

Multiple Grants Per Interval provides the ability to map multiple gates (application flows) using identical UGS traffic profiles destined for the same CM into a single flow at the DOCSIS level (Service Flow). This is accomplished by incrementing the number of grants per interval for each application flow serviced by a single DOCSIS service flow, resulting in multiple grants per interval (MGPI). This specification provides support for two approaches to MGPI, Flow-Aggregated and One-Flow. Flow-Aggregated MGPI is inherent in this specification and allows the AM to use the DOCSIS UGS traffic profile to explicitly set the number of grants per interval and place

several application flows on a single Gate. This results in an aggregated view for Event Messages, volume and time usage limits and opaque data. Due to the inherent nature of this capability, its use is not outlined in this section, but rather left to the AM to manage accordingly. One-Flow MGPI allows the CMTS to choose when to invoke MGPI independently of how the AM requests QoS. As far as the AM and PS are concerned, the single application flow per GateID mapping is preserved, while a single underlying DOCSIS resource is used for multiple gates with identical traffic characteristics. This allows for each application flow to have its own Event Messages, volume and time usage limits and opaque data while using a single DOCSIS Service Flow.

It is optional for a CMTS to support this feature. If the CMTS chooses to support MGPI functionality, it MUST follow the procedures outlined in the following sections. A CMTS SHOULD NOT attempt to invoke MGPI across services types (e.g., merge PacketCable Multimedia and PacketCable DqoS application flows together) as the interactions are not fully understood at this time and it may result in undesired operation.

Upon receipt of a Gate-Set without a gateID, the CMTS MUST first satisfy the requirements specified in Section 6.5.3. Once the resource envelops have been verified, the CMTS MAY compare the requested resources against those currently reserved or committed for the CM serving the SubscriberID provided in the Gate-Set. If a FlowSpec traffic profile is provided, the CMTS MUST translate the FlowSpec parameters to DOCSIS level parameters to make this comparison. If the requested QoS parameters (reserved and/or committed) do not match all of the QoS parameters for any of the existing service flows and resources are available to support the request, the CMTS MUST follow the requirements specified in Section 7.5.4 for creating new gates. If all the QoS parameters match and resources are available to support the request, the CMTS MAY modify the existing DOCSIS Service Flow by incrementing the grants per interval by one for a traffic profile described by a FlowSpec (or by the grants per interval value provided in a DOCSIS UGS object) and by adding the classifier(s) provided in the Gate-Set per the requirements in Section 9.3.4.1. Upon successful completion of the actions requested in the Gate-Set (e.g., modification of an existing DOCSIS Service Flow by adding the classifiers and incrementing the grants per interval associated with the Gate) the CMTS MUST respond with a Gate-Set-Ack. The CMTS MUST NOT respond with a Gate-Set-Ack until it has completed sufficient steps to ensure that any subsequent requests to Admit or Commit the Gate will not fail due to a lack of resources.

Once MGPI has been invoked (a second gate has been created using an existing Service Flow), the traffic profile cannot be changed by either Gate. If an update to one of the Gates is received which requires changes to the DOCSIS QoS parameters, the CMTS MUST create a new service flow and remove the application flow from the previously existing service flow. If only a single Gate is associated with a service flow, changes to the QoS parameters are allowed without creating a new service flow per the requirements in Section 6.5.6.

When a Gate is removed that is serviced with MGPI, the CMTS MUST update the DOCSIS service flow by removing the classifiers and decrement the grants per interval associated with the deleted Gate. In the case where the Gate being removed is the only flow (admitted or active), the entire DOCSIS service flow MUST be deleted as outlined in Section 6.5.8.

The requirements provided in Section 6.5.7.1 MUST be followed except as outlined in this section. When MGPI is invoked, the CMTS MUST maintain separate octet counters for each Gate associated with the Service Flow. This can be accomplished by using the packet counter for each classifier associated with a Gate and then multiplying the packet count by the packet size defined by the UGS flow to determine the octet count. It is important to acknowledge that in some cases this may not result in exact octet counts, while a UGS flow is deterministic, if an AM has reserved more bandwidth (larger packet size) then is actually used by the application, then the octet count will reflect the amount of traffic committed to cross the flow vs. the actual. Given this potential, if the CMTS receives a Gate-Set with a volume based usage limit it SHOULD service this gate with its own DOCSIS service flow to ensure accurate volume tracking is provided.

The One-Flow model also has the following limitations that should be understood, in particular the ability to track active flow time and set inactivity time-out limits per application flow is lost. While multiple sub-flows exists, if a single flow becomes inactive it will not be flagged by the CMTS as long as the other flows continue to transmit traffic. As a result, hung gates could result and persist until all other sub-flows are removed and the activity timer is triggered.

It is also noted that in addition to the limitations outlined above, the Flow-Aggregated model also loses the following capabilities: the ability to fully track each individual flow's reserved-to-committed state transitions, track byte counts per flow, track active flow time and set inactivity time-out limits per application flow. In DOCSIS this

level of tracking and control granularity is only available when a single traffic flow is defined for a service flow. So a decision to utilize flow-aggregation is a decision to not access these functions.

6.5.12 Procedures for Identifying a PDP to a PEP

When a PDP establishes a new connection with a PEP, it MAY send a PDP-Config message to declare the AMID(s) (for an application manager) or the PSID (for a policy server) that will be used by that PDP. This allows the PEP to associate state for this PDP from previous TCP connections with this new connection. For example, this allows a PEP to deliver Gate-Report-State messages on a new connection for a gate that may have been created using a connection that is no longer available.

A PDP is not required to send a PDP-Config message. However, if a PDP-Config message is sent then it MUST be the first message sent on a newly established connection once the connection initialization sequence described in Section 6.5.1 is complete. A PDP MUST NOT send more than one PDP-Config message on a single TCP connection. If a PDP does not send a PDP-Config message for a TCP connection, then it MUST NOT send any state synchronization requests on that connection. If a PEP receives a message that violates any of these rules it MUST send back the appropriate error message with an error code of "Unauthorized PSID" (if the PEP is a CMTS) or "Unauthorized AMID" (if the PEP is a PS).

When a PS receives a PDP-Config message from an AM it should record the association between the TCP connection and the AMID's in the message. The AM may issue requests using an AMID that has not been associated with that TCP connection through the PDP-Config message for the connection. When this happens, the PS should assume this is an implicit declaration of an association between the new AMID and that TCP connection. An implicit association should have the same behavior as an explicit association made with the PDP-Config message.

When a CMTS receives a PDP-Config message from a PS it should record the association between the TCP connection and the PSID in the message. For all subsequent Gate Control operations on that TCP connection the PSID is implicitly associated with the gate as part of the operation

6.5.13 Procedures for Delivering Gate-Report-State Messages

To determine how to send a Gate-Report-State message (or any other asynchronous notification message), a PEP should follow these steps (in order):

1. If the PEP has an association between the gate and an active TCP connection (such as the COPS Client Handle) then it should send the Gate-Report-State using that connection.

2. If the PEP is a CMTS, then it should identify all the active PDP connections that have an association with the PSID for the gate in the Gate-Report-State. If the PEP is a PS, then it should identify all the active PDP connections that have an association with the AMID for the gate in the Gate-Report-State. The PEP can send the Gate-Report-State on any of these connections.

3. If the PEP has an association between the IP Address of the connection originally used to create the gate and an active connection, then the PEP MAY use that connection to send the Gate-Report-State message.

4. If there is no such connection then the report can be dropped (Note: If the PEP supports incremental state synchronization the PEP should retain the information in the Gate-Report-State).

6.5.14 Procedures for Gate Control Operations Initiated by a Policy Server

There may be situations when it is desirable for a Policy Server to initiate a gate control operation (Gate-Info, or Gate-Delete) for gates that it may be "managing". For example, during a state synchronization a PS may need to issue a Gate-Info request to obtain the complete data for a gate. To accomplish this, the PS issues an operation as if it were sent from an AM, but it adds a PSID object to the message.

For PS-initiated operations the PSID in the request MUST be the same as the PSID specified in the PDP-Config message for the connection. If the PSID in a request does not match the PSID for the connection then the PEP should return the "Unauthorized PSID" error. If a Policy Server is going to issue PS-initiated operations, then it MUST send the PDP-Config message as described in Section 6.5.12. The TransactionID in a PS-initiated request MUST be unique among all the outstanding PS-initiated operations submitted to the same CMTS.

If the PSID object is present in a message, then it indicates that the operation being performed is being initiated by the Policy Server instead of the AM. Since the AMID/TransactionID tuple is not unique for PS-initiated operations, the CMTS should use the PSID/TransactionID tuple as a unique transaction identifier if one is needed.

When the response to a PS-initiated operation (either an Ack or Err) is sent from the CMTS, the PSID from the original request MUST be included in the response. If a PSID was not specified in the original request, then the associated reply MUST NOT include a PSID. This allows the Policy Server to distinguish responses to PS-initiated requests from other responses that must be forwarded to an AM.

6.5.15 Procedures for State Synchronization

This section provides a general overview of the synchronization process.

6.5.15.1 Initiating State Synchronization

Synchronization is always initiated by the PDP. To initiate synchronization, the PDP sends a Synch-Request message to the PEP. Although both the PSID and the AMID are optional in this message, one of the two objects MUST be present in the message. Both objects MAY be specified. If the PSID is specified, it MUST be the same value as the PSID specified in the PDP-Config message for the connection. If the PSID does not match, the PEP MUST return a Synch-Complete message with a PacketCable Error object with the "Unauthorized PSID" error.

As is the case with other gate control messages, if the PSID is specified in the request then it represents a PS-initiated request and the PSID/TransactionID tuple should be used as a unique transaction identifier. Otherwise the request is an AM-initiated request and the AMID/TransactionID tuple should be used as a unique identifier for the transaction.

Unlike other gate control operations this transaction can consist of multiple replies and is not finished until a Synch-Complete message is received by the PDP. All responses to a Synch-Request MUST be sent on the same TCP connection used to initiate the request. If that connection is terminated before the Synch-Complete message is received then the PDP MUST consider the synchronization to have terminated abnormally. It MAY re-issue the synchronization request when a new connection is established.

The PSID, AMID, and SubscriberID values that are specified in the Synch-Request are used as filtering criteria. The PEP MUST only include gates in the synchronization that have associations with all the specified values.

The Synch Options object MUST be included in the Synch-Request and is used to specify the type of synchronization to be performed, as well as the type of information to be sent back in the synchronization reports.

6.5.15.2 Full Synchronization

A full synchronization is one in which the PEP sends information about all the gates that match the filtering criteria (PSID, AMID, SubscriberID) specified in the originating Synch-Request. A PDP requests a full synchronization by setting the Synch Type field to 0 in the Synch Options object sent in the Synch-Request.

A full synchronization only includes gates that are still active (Gate-State is Authorized, Reserved, Committed, or Committed-Recovery)—it does not include any gates that have been closed or deleted.

All PEP's MUST support full synchronization.

A PDP MAY request a full synchronization at any time. Since a full synchronization may require the PEP to send information about many gates this operation can be very resource-intensive and should not be used unnecessarily.

6.5.15.3 Incremental Synchronization

An incremental synchronization is one in which the PEP sends information about gates that have had recent state changes for which the PDP may not have received notification. It is further restricted to gates that match the filtering criteria (PSID, AMID) specified in the Synch-Request. An incremental synchronization request cannot include the SubscriberID as filtering criteria. A PDP requests an incremental synchronization by setting the Synch Type field to 1 in the Synch Options object sent in the Synch-Request.

An incremental synchronization should only be performed after a new TCP connection has been established with a PEP. The purpose of the incremental synchronization is to provide the PDP with information that may have been

lost (or missed) while the connection was down. An incremental synchronization is limited to gates that have had Gate-Report-State messages whose delivery to the PDP were not confirmed. If the connection between a PDP and PEP has not been interrupted recently, then an incremental synchronization request should return no data.

An incremental synchronization may include gates that are no longer active. For example, if a gate was deleted because a timer expired but the associated Gate-Report-State message was undeliverable then that gate MUST be included in the incremental synchronization even though it no longer exists.

A PEP MAY support incremental synchronization. If a PEP receives a request for incremental synchronization and it does not support this feature, then it MUST return a Synch-Complete message with a PacketCable Error object with the error code set to "Unsupported Synch Type".

6.5.15.4 Reporting Gate State Information

The PEP sends one Synch-Report message to the PDP for each gate that is included in the synchronization. The TransactionID in the Synch-Report message MUST be the same as the TransactionID in the Synch-Request message that initiated the synchronization. The PSID in the Synch-Report MUST be the same value that was specified in the Synch-Request message. If there was no PSID in the original Synch-Request then there MUST NOT be a PSID in the associated Synch-Report messages. All other objects in the Synch-Report MUST be set to the values that are associated with the gate.

The Synch-Report message(s) MUST be sent on the same TCP connection on which the Synch-Request was received. If the connection is terminated before the synchronization is completed, any remaining Synch-Report messages MUST be dropped.

There are two variations on a Synch-Report. A "standard report" is limited to the set of objects that are normally provided in a Gate-Report-State message for a gate. To request standard reports the PDP sets the Report-Type field to 0 in the Synch Options object in the Synch-Request.

If the PDP needs more information about a gate than is provided in the standard report, it may issue a Gate-Info request to retrieve the complete definition of the gate. If the PDP knows in advance that it will need more information than is provided by the standard report for every gate included in the synchronization, then it can avoid issuing Gate-Info requests for every gate by requesting "complete reports" for each gate. To do this it sets the Report-Type field to 1 in the Synch Options object in the Synch-Request.

A "complete report" contains all the information that is normally provided in a Gate-Info-Ack message. In other words, the additional objects in the complete report include: GateSpec, Traffic Profile, Classifier(s), Event Generation Info, Volume-Based Usage Limit, and Time-Based Usage Limit.

Only "standard reports" can be returned for incremental synchronization requests. If a PEP receives a request for an incremental synchronization that also requests "complete reports", it MUST terminate the request by returning a Synch-Complete response with a PacketCable Error object with the error code for "Invalid Field for Object" and the error subcode specifying the Synch-Options object.

6.5.15.5 Completing the Synchronization

After all the Synch-Report messages for a synchronization request have been sent, the PEP MUST send a Synch-Complete message to the PDP to signal the end of the synchronization. The AMID and PSID objects MUST be specified as they were in the Synch-Request that initiated the synchronization. The TransactionID in the Synch-Complete message MUST be the same as the TransactionID in the Synch-Request message that initiated the synchronization.

The Synch-Complete message MUST be sent on the same TCP connection on which the Synch-Request was received. If the connection is terminated before the synchronization is completed, it should be dropped.

If an error occurs during the synchronization then the PEP MUST report the error by sending a Synch-Complete message with a PacketCable Error object. The Error Code should specify the reason for the error. The PacketCable Error object should only be included if there was an error processing the request.

There are several error codes needed to identify specific situations that may come up during a State Synchronization. These include:

- If the PEP has limited resources available for retaining incremental synchronization data and that limit is reached (causing it to discard some data), any subsequent incremental synchronization requests should fail with the "State Data Incomplete" error code.

- If the PEP is unable to process a synchronization request in a timely fashion because it is busy, it can return a Synch-Complete message with the "Insufficient Resources" error code. This does not indicate that the synchronization data is not available, only that the request could not be completed at the time it was issued. The PDP MAY re-issue the request at a later time.

- If a CMTS has rebooted and lost all of its previous state, it must convey this information to any PS that requests an incremental synchronization. If it were to simply respond to the PS indicating that it did not have any incremental changes then the PS might assume that any previously created gates still existed but this assumption is not valid in this situation. Therefore, if a CMTS receives an incremental synchronization request from a PS and the CMTS has not created a gate with the PSID for that PS since the CMTS was last rebooted (or it's state was last initialized) then the CMTS MUST return the "No State for PDP" error code. This is an indication to the PS that it must delete all state that was previously associated with this CMTS.

- Note that in the analogous situation where an AM sends an incremental synchronization request to a PS that has rebooted (and lost its previous state) the PS also MUST return the "No State for PDP" error code. However, in this situation the AM should not assume that previously existing gates have been deleted (because they may still exist in the CMTS). In this situation this message indicates to the AM that the PS is uncertain whether some incremental changes may have been lost and the AM should issue a full synchronization request in order to accurately determine the state of any previously existing gates in the CMTS.

6.5.15.6 Relaying Out of Sync Data

If a Policy Server discovers during the synchronization process that it is out of sync with a CMTS and that the associated data has not yet been forwarded to the Application Manager, then it MUST generate a Gate-Report-State for the affected gates and send those reports to the appropriate AM's. Creating a Gate-Report-State from a Synch-Report should be straightforward since the Synch-Report contains all the objects that are needed in a Gate-Report-State.

6.5.15.7 Synchronization with Stateless Policy Servers

Although a stateless policy server has no state, it should still support state synchronization. When an Application Manager issues a synchronization request it should be transparent to the AM whether the policy server maintains state. This may be implemented by:

- Translating incoming synchronization requests into synchronization requests to one (or more) of the CMTS's with which it is communicating. If a CMTS that is needed to complete the request does not support state synchronization (for example, if it is an I02 CMTS) then the PS MUST send an error response to the AM with error code "State Data Incomplete".

- Based on those requests it could translate and forward Synch-Report responses back to the AM.

- If a CMTS is unreachable or a CMTS returns an error then the PS would forward an error response back to the AM with error code "Transport Error" or "State Data Incomplete".

- If all the required CMTS's are reachable and the requests complete successfully, then the synchronization with the AM completes successfully. Note that in this situation the PS MUST insure that the originating AM receives only one Synch-Complete message.

Note: If a stateless policy server loses its TCP connection with a CMTS, there may be some situations where the PS may not be able to detect a potential loss of data. For example, if the CMTS does not support incremental synchronization the PS will not be able to determine if any Gate Report State messages may have been missed.

6.5.16 Procedures for Confirming PDP Receipt of Messages

Under some circumstances it may be desirable for a PEP to be able to determine if a message has actually been received by a PDP. For example, if a PEP implements incremental synchronization, it needs to determine if the PDP has actually received a Gate-Report-State message (Note: Just because it was sent successfully that does not mean that it was received).

To accomplish this, the PEP may add a Msg Receipt Key to any message sent from the PEP to the PDP. When the PDP receives a Msg Receipt Key, it MUST send a Msg Receipt message back to the PEP using the same TCP connection. By sending back the Msg Receipt Key, the PDP confirms that it has received the message that included that key as well as any other messages that preceded it.

Note: The PDP MUST send a Msg Receipt message for each message it receives with a Msg Receipt Key. It should send the Msg Receipt as soon as it receives the message; it should not delay sending the Msg Receipt until it has finished processing the message because the PEP may have resources associated with the request. If a PDP receives a message that is invalid (such as a message that cannot be parsed or that does not contain all the required objects), then the PDP MAY silently drop such a message without sending a Msg Receipt.

The Transaction Identifier in a Msg Receipt message is unused and MUST be set to 0. If there are any errors processing a Msg Receipt message the PEP MUST silently drop the message.

If a PDP does not send a PDP-Config message for a TCP connection, then the PEP MUST NOT send any Msg Receipt Key objects in any messages on that connection.

The specific contents of the Msg Receipt Key are left to the PEP implementation. It may use any value that best suits the PEP implementation and the PDP should make no assumptions about the value. For example, the PEP may choose to retransmit unconfirmed Gate-Report-State messages after a certain period of time. Or, the PEP may use this to implement an algorithm for tracking delivery in support of incremental synchronization. For example:

- it could send a Msg Receipt Key in every message it sends to a PDP, or in every n-th message.

- it could send a Msg Receipt Key based on a periodic interval.

- it could send a Msg Receipt Key if it has a buffer for storing unconfirmed incremental changes and that buffer is approaching its capacity.

6.5.17 Procedures for Indicating Updated or Lost Shared Resource

When Multicast Gates are in use, it is possible that changing conditions may make a single shared resource previously used for several Multicast Gates inappropriate for some or all of those Gates. In such a case, the CMTS MUST send a Gate-Report-State message to the Policy Server for each affected Gate indicating the change, and the Policy Server MUST in turn forward the message to the Application Manager.

If the shared resource can be replaced by another instance, then the CMTS MUST include Reason value 14 (Gate state unchanged, but SharedResourceID updated) and MUST include a new SharedResourceID value in the message.

If the shared resource is no longer available and cannot be replaced, then the CMTS MUST include Reason value 15 (close initiated by CMTS due to change of shared resource) and close the associated Gate for each affected Gate/Subscriber.

7 EVENT MESSAGING INTERFACE DESCRIPTION

7.1 Introduction

As in the PacketCable 1.x architecture, event messages within PacketCable Multimedia provide detailed information regarding QoS resource utilization, such as reservation, activation, and release. New for the PacketCable Multimedia framework is the need to track status of policy decisions (requests, updates, deletions). Also, since use of network resources falls outside the profiles within PacketCable 1.x (constant usage over time), there is the need to report volume-based and time-based usage information.

Event messages, as defined in this framework, are generated by network elements and stored on the Record Keeping Server (RKS). These EMs are then correlated by the RKS or other back-office system to record a single instance of a service. These records may be used to derive service billing information, network resource usage patterns, capacity planning, etc. EMs are not, however, intended for fault monitoring.

Currently, only the CMTS and Policy Server, which are part of the MSO's network and considered trusted entities, generate EMs within the Multimedia framework. Other elements of the network, such as the various client types, are considered untrusted. In the case of the Application Manager, this element may or may not be part of the MSO's network, and hence does not directly provide EMs to the RKS. The AM may, however, provide supplementary information as part of opaque data fields to the PS which would then be included in EMs generated by the PS.

PacketCable event messages for Multimedia represent a simplification and modification of PC 1.x event messages. Telephony-specific events such as Call_Answer and Call_Disconnect are considered optional, as are telephony service-specific event messages (e.g., Service Instance y). The intent is to leverage existing EM implementations as much as possible while providing sufficient abstraction mechanisms to support general Multimedia services.

Specifically, of the fourteen EM message types defined in support of PacketCable 1.x voice services, four will be required in PacketCable Multimedia, including QoS_Reserve, QoS_Commit, QoS_Release, and Time_Change. Three new EM message types relating to policy decisions are defined; Policy_Request, Policy_Delete, and Policy_Update. Table 6 below provides a summary overview of the PacketCable Multimedia EM message types.

Table 6 - PacketCable Multimedia EM Message Types

Event Message ID	Event Message	Originating Element	Description
7	QoS_Reserve	CMTS	Indicates the time at which the CMTS reserves bandwidth on the PacketCable access network. The CMTS must also generate this event if the reserved bandwidth changes.
8	QoS_Release	CMTS	Indicates the time at which the CMTS released its bandwidth commitment on the PacketCable access network.
17	Time_Change	PS, CMTS	Captures an instance of a time change. Whenever the (PacketCable) clock on a trusted network element (PS, and CMTS) is changed by more than 200 milliseconds, the network element MUST generate a Time_Change message.
19	QoS_Commit	CMTS	Indicates the time at which the CMTS commits bandwidth on the PacketCable access network. The CMTS must also generate this event if the committed bandwidth changes.
31	Policy_Request	PS	Indicates the time at which the Policy Server receives a new policy request from the AM.
32	Policy_Delete	PS	Indicates the time at which the Policy Server deletes a policy.
33	Policy_Update	PS	Indicates the time at which the Policy Server receives a request to update a policy.

Although PacketCable Multimedia event messages are based upon PacketCable 1.x, telephony specific events are optional for PacketCable Multimedia and are listed below. For further details on these events and associated attributes see the PacketCable 1.x EM specification [DOCSIS MULPIv3.0].

Table 7 - PacketCable 1.x Telephony EM Message Types

Event Message ID	Event Message	Description
1	Signaling_Start	Indicates the time at which signaling starts.
2	Signaling_Stop	Indicates the time at which signaling terminates.
3	Database_Query	Indicates the time at which a one-time request/response transaction or database dip is completed by an intelligent peripheral (e.g., 800 number database, LNP database).
6	Service_Instance	Indicates the time at which the CMS provides an instance of a call control/feature service (e.g., call hold, call waiting).
9	Service_Activation	Indicates the time at which the CMS records an attempt to activate a service (e.g., call forwarding, call waiting).
10	Service_Deactivation	Indicates the time at which the CMS records an attempt to deactivate a service (e.g., call forwarding, call waiting).
13	Interconnect_Start	Indicates the time at which the start of network interconnect signaling occurs.
14	Interconnect_Stop	Indicates the termination of bandwidth between the PacketCable network and the PSTN.
15	Call_Answer	Indicates that the media connection is open because an answer event has occurred.
16	Call_Disconnect	Indicates the time at which the media connection is closed because the calling party has terminated the call by going on-hook, or that the destination party has gone on-hook and the called-party's call-continuation timer has expired.
20	Media_Alive	Indicates that service is active due to the continued existence of a bearer connection. This message may be generated by any trusted PacketCable network element (CMS, MGC, and CMTS) as the vendor sees fit.

7.2 Record Keeping Server Requirements

The Record Keeping Server (RKS) is a trusted network element function. The RKS is generally depicted in this specification as a distinct standalone element, but this specification does not preclude some other application from performing the functions of an RKS, providing that the application conforms to the requirements herein.

The RKS is the mediation layer between the PacketCable Multimedia network and the back-office applications. The RKS is expected to process the data received from the PacketCable Multimedia network and to present it to the back-office applications in the format and within the time constraints deemed necessary by the MSO. The RKS therefore acts as a demarcation point between the PacketCable network and the back office applications.

The RKS MUST be capable of receiving and processing Event Messages formatted in accordance with this specification.

The RADIUS messages inside which Event Messages are encapsulated are transported over UDP, which does not guarantee reliable delivery of messages; hence the request/response nature of the protocol defined herein. When an RKS receives and successfully records all the PacketCable Event Messages contained in a RADIUS Accounting-Request message, it MUST transmit an Accounting-Response message to the client. The RKS MUST NOT transmit an Accounting-Response reply if it fails to record successfully all the Event Messages in a RADIUS Accounting-Request message.

The RKS SHOULD ignore Event Messages where the PacketCable "Event Message type" is unrecognized. The RKS SHOULD also ignore PacketCable event attributes where the Event Attribute ID is unrecognized.

7.3 General PacketCable Multimedia Network Element Requirements

This section lists requirements placed on the PacketCable Multimedia network elements.

7.3.1   Element ID

Each PacketCable network element that generates an Event Message MUST identify itself with a static, unique element ID. The Element ID is a statically configured element number, unique within a PacketCable domain, which MUST be in the range 0 to 99,999.

7.3.2   Timing

It is important for elements that generate Event Messages to remain closely synchronized with one another and with a standard clock. The requirements in this section ensure that such elements maintain this synchronization and report events with timestamps that are both accurate and precise.

Elements that generate Event Messages MUST use the Network Time Protocol as defined in [DOCSIS MULPIv3.1]. Elements MUST operate in mode 3 (Client mode). The value of NTP.MAXPOLL MUST NOT exceed eleven, which corresponds to 2048 seconds.

Event Messages MUST include timestamps with a precision of one millisecond.

7.3.3   Primary and Secondary RKS Considerations

PacketCable Multimedia supports an architecture that consists of a primary and secondary RKS. The secondary RKS is used as a fallback RKS when a network element (PS, CMTS) is unable to successfully send a message to the primary RKS. PacketCable Multimedia network elements MUST support event message transport to a primary RKS and failover to a secondary RKS when communication with the primary RKS fails. Once a network element fails over to the secondary RKS, the secondary becomes the primary for the duration of that session or gate. The Policy Server is provisioned with primary and secondary RKSes as required for the applications it supports. The PS MUST provide the IP address and port of the primary RKS and optionally the secondary RKS to the CMTS in policy decision messages (Gate-Set). The PS MUST support multiple sets of primary and secondary RKSes.

In order to guarantee the reliable transfer of the data the network elements should implement a user configurable retry time interval and the number of times the client needs to retransmit the event. The time interval should be configurable (suggested: 10ms to 10 s), the number of retries should be configurable (suggested: 0 to 9). The number of retries should be attempted on both the primary RKS and secondary RKS. After exhausting the number of retries the event message should be written to an error file, and the event message can then be deleted from the network element.

If the PacketCable network element does not receive an Accounting-Response within the configured retry interval, it MUST continue resending the Accounting-Request until it receives an Accounting-Response from an RKS or the maximum number of retries is reached. The PacketCable network element MUST re-send the same Accounting Request to the primary RKS and if the retry limit is reached, re-send the same Accounting Request to the secondary RKS.

All Network Elements MUST store Event Messages until they have received an Acknowledgement (Accounting Response) from an RKS that the data was correctly received and stored, or until the maximum number of retries has been reached. Only when an Ack is received or the maximum retries reached are the network elements allowed to delete these Event Messages.

Once a Network Element succeeds in sending event messages to the secondary RKS, a failover to the secondary RKS occurs. This is a non-revertive failover, meaning that the secondary RKS becomes active, and is the new primary RKS. All subsequent event messages for the session MUST be sent to the now active secondary RKS. For all new sessions, the PDP MUST instruct the PEP to use the new active RKS as the primary (i.e., the previous secondary RKS becomes the new primary for subsequent session). Note that it is possible under certain circumstances that one element, PS or CMTS, may be able to communicate with the primary RKS while the other element may not for the same session. In cases such as this it is expected that the RKS be able to reconcile event messages between the primary and secondary RKS.

7.3.4   Interaction with PacketCable RKS

An Application Manager MAY provide an optional Event Generation Info object in the first Gate-Set message to enable Event Messaging for the Gate. If the Event Generation Info object is not provided in the first Gate-Set

message (by either the AM or PS), Event Messaging MUST not be used for the life of the Gate. In either case, all subsequently supplied Event Generation Info objects for this Gate MUST be ignored by the PEP.

If present, this object MUST contain a valid BCID which can be used by the AM, PS, and CMTS to correlate billing information for the flow. If an Application Manager provides a BCID to the PS and the AM is trusted by the PS, the PS MAY use the BCID provided by the Application Manager.

If an Application Manager provides an optional Event Generation Info object which specifies primary and secondary RKS IP Addresses that the Policy Server does not have knowledge of, the Policy Server MUST send the Event Messages for that transaction to the default primary RKS IP Address except in failover circumstances in which case the Event Messages MUST be sent to the default secondary RKS IP Address.

The Application Manager MAY specify a primary RKS IP address in the optional Event Generation Info object or the Application Manager MAY allow the Policy Server to use its default primary and secondary RKS IP Addresses. If the AM specifies a primary RKS IP Address, it MAY also specify a secondary RKS IP Address. The Application Manager indicates that an RKS is not specified by setting the primary and secondary RKS IP address and port to zero.

Regardless of what the Policy Server may receive from the Application Manager, the PS MUST direct the CMTS to use the same BCID and Primary/Secondary RKS IP Addresses and Ports which the Policy Server chooses to use. The PS should decide which RKS pair to send to based on AMID.

7.4   Event Messages for PacketCable Multimedia

This section provides a detailed description and definition of each of the Event Messages defined for PacketCable Multimedia.

7.4.1   Policy Events

The Policy Event Messages are new for PacketCable Multimedia. They indicate the time at which the Policy Server receives a request for a policy action and serve to bracket the ensuing set of Event Messages for any resource usage associated with the various instances of a service. Policy event messages are used to indicate the initial policy request, an update to the policy, and the deletion of a policy.

The PS MUST timestamp the Policy Event Messages upon receiving a policy request message from the AM. Immediately upon receiving an initial policy request, the PS MUST create a Billing Correlation ID (BCID). Each generated BCID MUST conform to the Billing Correlation ID (BCID) Attribute Structure format requirements in Table 18.

The PS MUST include the BCID in the EM header for all subsequently generated Policy Event Messages associated with this request. Also, the PS MUST include the BCID in the Gate-Set message sent to the CMTS.

The PS MUST generate policy event messages immediately after determining the result of a policy request. The result may be based upon PS internal authorization and admission control mechanisms or, upon receiving a response to its Gate-Set and Gate-Delete messages from the CMTS. The PS creates a timestamp for an event message when it receives a request from the AM but does not generate the event until it knows the outcome of the request. For Multicast Gates, the PS additionally indicates the Multicast Classifier and the SharedResourceID in the policy event messages.

7.4.1.1   Policy_Request

The Policy Server MUST send a Policy_Request event message to the RKS if a request to create a new policy is received. The PS MUST set the Policy_Decision_Status to either Approved (1) or Declined (2), based upon the outcome of authorization and admission control.

Note: Because the PS does not send the Policy_Request event message until after the CMTS responds to the Gate-Set message, it is possible that QoS event messages from the CMTS may arrive at the RKS prior to a Policy-Request event message.

Table 8 - Policy_Request Event Message

Attribute Name	Required or Optional	Comment
Event_Message_Header	R	See Table 17
Application_Manager_ID	R	Contains the network wide unique Application Manager Tag of the AM and Application Type
Subscriber_ID	O	If the Gate-Set message contains the SubscriberID object, the PS MUST include this in the Event Message.
IPv6Subscriber_ID	O	If the Gate-Set message contains the IPv6SubscriberID object, the PS MUST include this in the Event Message.
Policy_Decision_Status	R	1 – Policy Approved 2 – Policy Denied
Policy_Denied_Reason	O	Required when Policy_Decision_Status = 2 (Policy Denied) This MUST be set to the Error-Code from the PacketCable Error object in the corresponding Gate-Set-Err message, as defined in Section 6.4.2.14.
FEID	R	Financial Entity ID. Identifies the paying entity. Supplied by the PS.
AM_Opaque_Data	O	If the AM includes the Opaque Data object in the Gate-Set, then the PS MUST include this in the Event Message.
Volume_Usage_Limit	O	If the AM includes the Volume-Based Usage Limit object in the Gate-Set, then the PS MUST include this in the Event Message.
Time_Usage_Limit	O	If the AM includes the Time-Base Usage Limit object in the Gate-Set, then the PS MUST include this in the Event Message.
User_ID	O	If the AM includes the UserID object in the Gate-Set, then the PS MUST include this in the Event Message.
Multicast Classifier	O	Only included for Multicast Gates, if the AM includes a classifier in the Gate-Set to a multicast destination address, then the PS MUST include this in the Event Message.
SharedResourceID	O	Only included for Multicast Gates, this is the Id returned by the CMTS on a successful Gate-Set, the PS MUST include this in the Event Message.

7.4.1.2 Policy_Delete

The Policy Server MUST send a Policy_Delete Event Message to the RKS when it receives a Gate-Delete-Ack from the CMTS in response to an AM or PS initiated Gate-Delete, or a Gate-Report-State from the CMTS indicating that the resources are no longer available for a session. The PS MUST always generate a Policy_Delete Event Message to close a session if it has previously generated a Policy-Request Event Message to open the session.

Table 9 - Policy_Delete Event Message

Attribute Name	Required or Optional	Comment
Event_Message_Header	R	See Table 17
Application_Manager_ID	R	Contains the network wide unique Application Manager Tag of the AM and Application Type
Subscriber_ID	O	If the Gate-Delete-Ack or Gate-Report-State message contains the SubscriberID object, the PS MUST include this in the Event Message.
IPv6Subscriber_ID	O	If the Gate-Delete-Ack or Gate-Report-State message contains the IPv6SubscriberID object, the PS MUST include this in the Event Message.
Policy_Deleted_Reason	R	1 – Application Manager request 2 – CMTS decision 127 – Other
FEID	R	Financial Entity ID. Identifies the paying entity. Supplied by the PS.
AM_Opaque_Data	O	If the Gate-Delete-Ack or Gate-Report-State message contains the Opaque Data object, then the PS MUST include this in the Event

Attribute Name	Required or Optional	Comment
		Message.

7.4.1.3   Policy_Update

The Policy Server MUST send a PolicyUpdate Event Message to the RKS if a request to change the traffic profile, classifier, volume limit, time limit, or opaque data of a Gate is received from the AM. Additionally for Multicast Gates, if the PS receives an update to the SharedResourceID from the CMTS, the PS MUST send a PolicyUpdate Event Message to the RKS with the new SharedResourceID.

Table 10 - Policy_Update Event Message

Attribute Name	Required or Optional	Comment
Event Message Header	R	See Table 17
Application_Manager_ID	R	Contains the network wide unique Application Manager Tag of the AM and Application Type
SubscriberID	O	If the Gate-Set message contains the SubscriberID object, the PS MUST include this in the Event Message.
IPv6Subscriber_ID	O	If the Gate-Set message contains the IPv6SubscriberID object, the PS MUST include this in the Event Message.
Policy_Decision_Status	R	1 – Policy Approved 2 – Policy Denied
Policy_Denied_Reason	O	Required when Policy_Decision_Status = 2 (Policy Denied) This MUST be set to the Error-Code from the PacketCable Error object in the corresponding Gate-Set-Err message, as defined in Section 6.4.2.14.
Policy_Update_Reason	R	1 – Traffic Profile 2 – Classifier 3 – Volume Limit 4 – Time Limit 5 – Opaque data 6 – Multiple Updates (combination of 1-5) 7 – SharedResourceID Change 127 – Other
FEID	R	Financial Entity ID. Identifies the paying entity. Supplied by the PS.
AM_Opaque_Data	O	If the AM includes the Opaque Data object in the Gate-Set, then the PS MUST include this in the Event Message.
Volume_Usage_Limit	O	If the AM includes the Volume-Based Usage Limit object in the Gate-Set, then the PS MUST include this in the Event Message.
Time_Usage_Limit	O	If the AM includes the Time-Base Usage Limit object in the Gate-Set, then the PS MUST include this in the Event Message.
User_ID	O	If the AM includes the UserID object in the Gate-Set, then the PS MUST include this in the Event Message.
SharedResourceID	O	Only included for Multicast Gates, the PS MUST include this in the Event Message if the SharedResourceID has been updated. The PS MUST use Policy_Update_Reason 7 for this purpose.

7.4.2   QoS Events

The QoS Event Messages are sent by the CMTS to the RKS and indicate events related to the PacketCable access network resources used by the CMTS.

7.4.2.1   QoS_Reserve

This Event Message indicates the time at which the CMTS reserves bandwidth on the PacketCable access Network. The CMTS MUST also generate this event if the Reserved bandwidth changes.

The CMTS MUST timestamp this message immediately upon transmission of a DSA-ACK or DSC-ACK acknowledging a successful DSA-RSP or DSC-RSP to the CM which completes a transaction reserving resources.

If the DSA-RSP or DSC-RSP confirmation code from the CM is not successful, the CMTS MUST NOT generate this message.

The CMTS is not required to support Event Messaging for Multicast Gates and the message details are currently not defined. At this time the CMTS MUST NOT send a QoS_Reserve message for Multicast Gates.

For Upstream Drop traffic profiles the CMTS MUST set the SF_ID to 0.

Table 11 - QoS_Reserve Event Message

Attribute Name	Required or Optional	Comment
Event Message Header	R	See Table 17
QoS_Descriptor	R	None
SF_ID	R	None
Flow_Direction	R	None
Element_Requesting_QoS	R	0 = Client 1 = Policy Server 2 = Embedded Client

7.4.2.2   QoS_Commit

The QoS_Commit Event Message indicates the time at which the CMTS commits bandwidth on the PacketCable access Network. The CMTS MUST also generate this event if the Committed bandwidth changes.

The CMTS MUST timestamp this message immediately upon transmission of a DSA-ACK or DSC-ACK acknowledging a successful DSA-RSP or DSC-RSP to the CM which completes a transaction committing resources.

If the DSA-RSP or DSC-RSP confirmation code from the CM is not successful, the CMTS MUST NOT generate this message.

For Upstream Drop traffic profiles the CMTS MUST set the SF_ID to 0.

The CMTS is not required to support Event Messaging for Multicast Gates and the message details are currently not defined. At this time the CMTS MUST NOT send a QoS_Commit message for Multicast Gates.

Table 12 - QoS_Commit Event Message

Attribute Name	Required or Optional	Comment
Event Message Header	R	See Table 17
QoS_Descriptor	R	None
SF ID	R	Indicates the Service Flow ID for unicast Service Flows
Flow_Direction	R	None

7.4.2.3   QoS_Release

The QoS_Release Event Message indicates the time at which the CMTS releases its reservation and/or bandwidth commitment on the PacketCable access network.

The CMTS MUST timestamp this message immediately upon Transmission of a DSD-REQ that indicates the request to delete bandwidth.

The CMTS is not required to support Event Messaging for Multicast Gates and message details are currently not defined. At this time the CMTS MUST NOT send a QoS_Release message for Multicast Gates.

Table 13 - QoS_Release Event Message

Attribute Name	Required or Optional	Comment
Event Message Header	R	See Table 17
SF ID	R	None
Flow_Direction	R	None
QoS_Release_Reason	R	1 – Gate Closed by PS 2 – Inactivity resource recovery (T4) timer expiration 3 – CM Failure 4 – Pre-Empted 5 – RSVP PathTear request 6 – CM request 7 – Admitted (T2) timer expiration 127 – Other
Gate_Usage_Info	R	None
Gate_Time_Info	R	None

7.4.3   Time_Change

This event captures an instance of a time change. Whenever the (PacketCable) clock on the network element (PS or CMTS) is changed by more than 200 milliseconds, the network element MUST generate a Time Change message. This includes time shift events (Daylight savings time), step adjustments to synchronize with the NTP reference clock and manual time setting changes. The Event_Time attribute in the Event Message header MUST reflect the new (adjusted) notion of time. Note that Time_Change message is not required for slew adjustments performed by NTP.

The network element (PS and CMTS) MUST send the Time Change event message to the active (current primary) RKS. The Time Change event message MUST be generated when a gate(s) is currently present in the CMTS. The Time Change event message need not be generated when there are no gates in the CMTS. Only one Time Change event message is sent to each primary RKS regardless of how many gates may exist on the CMTS. In other words, if the CMTS has several gates all of which point to the same RKS, then only one Time Change event message should be sent to that RKS.

The BCID in the Event Message Header of the Time Change event message MUST be generated locally by the network element at the time of the event. The BCID is not associated with any session related BCID, it is a unique BCID for this event.

Table 14 - Time_Change Event Message

Attribute Name	Required or Optional	Comment
Event Message Header	R	See Table 17
Time_Adjustment	R	None

7.5   Event Messaging Attributes for PacketCable Multimedia

This section describes and defines the PacketCable attributes included in the PacketCable Event Messages.

Table 15 provides a mapping between each of the PacketCable Event Messages and their associated attributes. Table 16 offers a detailed description of each of these attributes.

Table 15 - PacketCable Attributes Mapped to PacketCable MM Event Messages

EM Attribute ID	EM Attribute Name	7 – QoS_Reserve	8 – QoS_Release	17 – Time_Change	19 – QoS_Commit	31 – Policy_Request	32 – Policy_Delete	33 – Policy_Update
1	Event_Message_Header	X	X	X	X	X	X	X
30	SF_ID	X	X		X			
32	QoS_Descriptor	X			X			
38	Time_Adjustment			X				
49	FEID					X	X	X
50	Flow_Direction	X	X		X			
61	AM_Opaque_Data					X	X	X
62	Subscriber_ID					X	X	X
63	Volume_Usage_Limit					X		X
64	Gate_Usage_Info		X					
65	Element_Requesting_Qos	X						
66	QoS_Release_Reason		X					
67	Policy_Denied_Reason					X		X
68	Policy_Deleted_Reason						X	
69	Policy_Update_Reason							X
70	Policy_Decision_Status					X		X
71	Application_Manager_ID					X	X	X
72	Time_Usage_Limit					X		X
73	Gate_Time_Info		X					
74	IPv6Subscriber_ID					X	X	X
75	User_ID					X		X
77	Multicast Classifier					X	X	X
78	SharedResourceID					X	X	X

Table 16 provides a detailed definition of each of the PacketCable Event Message attributes. A data value of an attribute may either be represented by a simple data format (one data field) or by a more complex data structure.

Table 16 - PacketCable MM Event Message attributes

EM Attribute ID	EM Attribute Length	EM Attribute Name	EM Attribute Value Type	Attribute Data Description
1	76 bytes	EM_Header	Data structure See Table 17	Common data required on every PacketCable Event Message
30	4 bytes	SF_ID	Unsigned integer	Service Flow ID, a 32-bit integer assigned by the CMTS to each DOCSIS Service Flow defined within a DOCSISRF MAC domain. SFIDs are considered to be in either the upstream direction (USFID) or downstream direction (DSFID). USFIDs and DSFIDs are allocated from the same SFID number space.
32	Variable; Min 8 bytes	QoS_Descriptor	Data structure See Table 20	QoS parameters data

EM Attribute ID	EM Attribute Length	EM Attribute Name	EM Attribute Value Type	Attribute Data Description
38	8 bytes	Time_Adjustment	Signed integer	Time adjustment of an element's (PS, CMTS) clock. This time is in milliseconds, detailing the amount of the time change.
49	Variable length, maximum of 247 bytes	FEID	ASCII character string.	Financial Entity ID. The first 8 bytes constitute MSO defined data. By default, the first 8 bytes are zero filled. From the 9th byte on the field contains the MSO's domain name which uniquely identifies the MSO for billing and settlement purposes. The MSO's domain name is limited to 239 bytes.
50	2 bytes	Flow Direction	Unsigned integer	Flow direction: 0 = Reserved 1 = Upstream 2 = Downstream
61	8 bytes	AM_Opaque_Data	Unsigned integer	Opaque data passed from Application Manager.
62	4 bytes	Subscriber_ID	Unsigned integer	4 concatenated byte values representing an IPv4 address
63	8 bytes	Volume_Usage_Limit	Unsigned integer	Volume limit in octets set by the AM
64	8 bytes	Gate_Usage_Info	Unsigned integer	The number of octets transmitted on the DOCSIS RF network from the byte after the MAC header HCS to the end of the CRC
65	2 bytes	Element_Requesting_QoS	Unsigned integer	0 = Client 1 = Policy Server 2 = Embedded Client
66	2 bytes	QoS_Release_Reason	Unsigned integer	1 – Gate Closed by PS 2 – Inactivity resource recovery (T4) timer expiration 3 – CM Failure 4 – Pre-Empted 5 – RSVP PathTear request 6 – CM request 7 – Admitted (T2) timer expiration 127 – Other
67	2 bytes	Policy_Denied_Reason	Unsigned integer	The Error-Code from the PacketCable Error object in the corresponding Gate-Set-Err message, as defined in Section 6.4.2.14.
68	2 bytes	Policy_Deleted_Reason	Unsigned integer	1 – Application Manager request 2 – CMTS decision 127 – Other
69	2 bytes	Policy_Update_Reason	Unsigned integer	1 – Traffic Profile 2 – Classifier 3 – Volume Limit 4 – Time Limit 5 – Opaque data 6 – Multiple Updates (combination of 1-5) 127 – Other
70	2 bytes	Policy_Decision_Status	Unsigned integer	1 – Policy Approved 2 – Policy Denied
71	4 bytes	Application_Manager_ID	Unsigned integer	Application Manager ID, composed of Application Manager Tag and Application Type, as defined in Section 6.4.2.2.
72	4 bytes	Time_Usage_Limit	Unsigned integer	Time limit in seconds set by the AM
73	4 bytes	Gate_Time_Info	Unsigned integer	The number of seconds a gate has been in either the Committed or Committed-Recovery states
74	16 bytes	IPv6Subscriber_ID	Unsigned integer	16 concatenated byte values representing an IPv6 address

EM Attribute ID	EM Attribute Length	EM Attribute Name	EM Attribute Value Type	Attribute Data Description
75	Variable	UserID	UTF-8 character string	Character string identifying the user associated with reported Event.
76	Variable (24, 40, 64 bytes)	Multicast Classifier	Data structure See 6.4.2.6 Classifiers	Classifier information, required for Multicast Gates only. This is the Classifier, or Extended Classifier, or the IPv6 classifier object defined for a Multicast Gate.
77	4 bytes	SharedResourceID	Unsigned integer	The Shared Resource Identifier is a 4-byte unsigned integer value assigned by the CMTS only for Multicast Gates.

7.5.1 EM_Header Attribute Structure

Table 17 contains a detailed description of the fields in the EM_Header attribute structure. This Event Message Header attribute MUST be the first attribute in every PacketCable Event Message.

Table 17 - EM_Header Attribute Structure

Field Name	Semantics	Value Type	Length
Version_ID	Identifies version of this EM Header structure. 1 = PacketCable 1.0 2 = PacketCable 1.1 3 = PacketCable Multimedia Note: A value of 2 or 3 indicates the Event_Object field in this header is used. The PS and CMTS network elements MUST set the value of the Version_ID to 3.	Unsigned integer	2 bytes
BCID	Unique identifier for a transaction within a network.	Data Structure See Table 18	24 bytes
Event Message Type	Identifies the type of Event Message.	Unsigned integer	2 bytes
Element Type	Identifies Type of Originating Element: 0 = Reserved 1 = Reserved 2 = CMTS 3 = Reserved 4 = Policy Server	Unsigned integer	2 bytes
Element ID	Network wide unique identifier 5 digits (statically configured element number unique within a PacketCable domain in the range of 0-99,999)	Right justified, space padded ASCII Character String	8 bytes
Time Zone	Identifies daylight savings time and offset from universal time (UTC). Daylight Savings Time: 0 = Standard Time 1 = Daylight Savings The Daylight-Savings Time indicator MUST be set to a value of 1 if the network element is in a region that implements DST and then only during the daylight-saving-time period (usually the summer months). Since there may be areas in which the daylight-saving-time offset indicates a time-change other than 1 hour, the receiving system (e.g., RKS) needs to correctly calculate local time based on knowledge of the area(s) in which the subscriber and the network element reside. UTC offset: ± HHMMSS The offset is reported from the network element (PS, CMTS) point of view; not based on the subscriber point of view. The UTC offset represents the time offset from universal time (formerly called Greenwich Mean Time, or GMT) when standard time is in effect and MUST NOT change on transition into or out of daylight-saving-time. For example: The Time_Zone field of a network element in Boston	ASCII character string	1 byte 7 bytes

Field Name	Semantics	Value Type	Length
	in December is "0-050000". The same network element in Boston in July is "1-050000".		
Sequence Number	Each network element MUST assign a unique and monotonically increasing unsigned integer for each Event Message sent to a given RKS. For the purpose of this specification, monotonically increasing is to be interpreted as increasing by 1. This is used by the RKS to determine if Event Message are missing from a given network element.	Unsigned integer	4 bytes
Event_time	Event generation time and date. Millisecond granularity. This specifies the local time. i.e., after applying Time_Zone UTC offset and Daylight Savings Time adjustment to UTC time. Format: yyyymmddhhmmss.mmm	ASCII character string	18 bytes
Status	Status indicators	See Table 19	4 bytes
Priority	Indicates the importance to assign relative to other event messages. 255 = highest priority 0 = lowest priority 128 = default.	Unsigned integer	1 byte
Attribute Count	Indicates the number of attributes that follow (or are appended to) this header in the current Event Message	Unsigned integer	2 bytes
Event_Object	The Event_Object field allows for a grouping of services. 0 = Accounting Event Message 1 = Reserved The PS and CMTS network elements MUST set the value of the Event_Object field to 0 if the EM_Header Version_ID is 3 (PacketCable Multimedia Event Message to the RKS). The RKS MUST discard EM messages when the Event_Object field is set to 1.	Unsigned integer	1 byte

7.5.2   Billing Correlation ID (BCID) Field Structure

Table 18 describes the Billing Correlation ID field (BCID). The RKS, or some other back office application, uses the BCID to correlate Event Messages that are generated for a single transaction. It is one of the fields in the Event Message Header Attribute. The BCID is unique for each transaction in the network. All Event Messages from the same network element with the same BCID MUST be sent to the same primary RKS except in failover circumstances in which case the Event Messages MUST be sent to secondary RKS.

Table 18 - BCID Field Description

Field Name	Semantics	Value Type	Length
Timestamp	High-order 32 bits of NTP time reference	Unsigned integer	4 bytes
Element_ID	Network wide unique identifier 5 digits (statically configured element number unique within a PacketCable domain in the range of 0-99,999)	Right justified, space padded ASCII character string	8 bytes

Field Name	Semantics	Value Type	Length
Time Zone	Identifies daylight savings time and offset from universal time (UTC).	ASCII character string	
	Daylight Savings Time:		
	0 = Standard Time 1 = Daylight Savings		1 byte
	The Daylight-Savings Time indicator MUST be set to a value of 1 if the network element is in a region that implements DST and then only during the daylight-saving-time period (usually the summer months). Since there may be areas in which the daylight-saving-time offset indicates a time-change other than 1 hour, the receiving system (e.g., RKS) needs to correctly calculate local time based on knowledge of the area(s) in which the subscriber and the network element reside.		
	UTC offset: ± HHMMSS		
	The offset is reported from the network element (PS, CMTS) point of view; not based on the subscriber point of view.		
	The UTC offset represents the time offset from universal time (formerly called Greenwich Mean Time, or GMT) when standard time is in effect and MUST NOT change on transition into or out of daylight-saving-time.		7 bytes
	For example: The Time_Zone field of a network element in Boston in December is "0-050000". The same network element in Boston in July is "1-050000".		
	For example: The Time_Zone field of a network element in Boston in December is "0-050000". The same network element in Boston in July is "1-050000".		
Event Counter	Monotonically increasing for each Transaction	Unsigned integer	4 bytes

7.5.3 Status Field Structure

The Status field of the Event Message Header Attribute is a 32-bit mask. Bit 0 is the low-order bit; the field is treated as a 4 byte unsigned integer. Table 19 presents Status field description.

Table 19 - Status Field Description

Start Bit	Semantics	Bit Count
0-1	Error Indicator: 0 = No Error 1 = Possible Error 2 = Known Error 3 = Reserved	2
2	Event Origin: 0 = Trusted Element 1 = Untrusted Element	1
3	Event Message Proxied: 0 = Not proxied, all data known by sending element 1 = proxied, data sent by a trusted element on behalf of an untrusted element	1
4-31	Reserved. The Status field bits 4 to 31 MUST be set to 0.	28

7.5.4 QoS Descriptor Attribute Structure

Table 20 describes the QoS Descriptor Data Structure.

Table 20 - QoS Descriptor Data Structure

Field Name	Semantics	Value Type	Length
Status_Bitmask	Bitmask describing structure contents (See Table 21)	Bit map	4 bytes

Field Name	Semantics	Value Type	Length
Service_Class_Name	Service profile name	Right justified, space padded ASCII character string	16 bytes
QoS_Parameter_Array	QoS Parameters. Contents determined by Status Bitmask	Unsigned integer array	Variable length array of 32-bit unsigned integers

Table 21 describes the QoS Status Bitmask field of the QoS Descriptor attribute. Bits 2-17 describe the contents of the QoS_Parameter_Array. Each of these bits indicates the presence (bit=1) or absence (bit=0) of the named QoS parameter in the array. The location of a particular QoS parameter in the array matches the order in which that parameter's bit is encountered in the bitmask, starting from the low-order bit.

Each QoS parameter present in the QoS_Parameter_Array must occupy four bytes. The definition and encoding of the QoS parameters can be found in Appendix C of the DOCSIS RFI specification. QoS parameters whose definition specifies less than four bytes must be right-justified (where the 4 bytes are to be treated as an unsigned integer) in the four bytes allocated for the array element.

Table 21 - QoS Status Bitmask

Start Bit	Semantics	Bit Count
0	State Indication: 0 = Illegal Value 1 = Resource Reserved but not Activated 2 = Illegal Value 3 = Resource Reserved & Activated	2
2	Service Flow Scheduling Type	1
3	Nominal Grant Interval	1
4	Tolerated Grant Jitter	1
5	Grants Per Interval	1
6	Unsolicited Grant Size	1
7	Traffic Priority	1
8	Maximum Sustained Rate	1
9	Maximum Traffic Burst	1
10	Minimum Reserved Traffic Rate	1
11	Minimum Packet Size	1
12	Maximum Concatenated Burst	1
13	Request/Transmission Policy	1
14	Nominal Polling Interval	1
15	Tolerated Poll Jitter	1
16	IP Type of Service Overwrite	1
17	Maximum Downstream Latency	1
18	Downstream Peak Traffic Rate	1
19	Provisioned Attribute Mask	1
20	Required Attribute Mask	1
21	Forbidden Attribute Mask	1
22	Attribute Aggregation Rule Mask	1
23	Upstream Peak Traffic Rate	1
24	Minimum Buffer	1
25	Target Buffer	1
26	Maximum Buffer	1

7.6   RADIUS Accounting Protocol

This section specifies the protocol used between the PacketCable network elements that generate Event Messages (PS, CMTS) and the Record Keeping Server (RKS). These network elements MUST support RADIUS Accounting [IETF RFC 2866] with PacketCable extensions as defined in this document.

The RADIUS Accounting protocol is a client/server protocol that consists of two message types: Accounting-Request and Accounting-Response. PacketCable network elements that generate Event Messages are RADIUS clients that send Accounting-Request messages to the RKS. The RKS is a RADIUS server that sends Accounting-Response messages back to the PacketCable network elements indicating that it has successfully received and stored the Event Message.

The Event Messages are formatted as RADIUS Accounting-Request and Accounting-Response packets as specified in [DOCSIS MULPIv3.0].

7.6.1   Authentication and Confidentiality

Refer to Section 8 for details concerning the use of IPsec to provide both authentication and confidentiality of the RADIUS messages, and the details of the correct usage of the RADIUS shared secret.

7.6.2   Standard RADIUS Attributes

Each RADIUS message starts with the standard RADIUS header shown in Table 22.

Table 22 - RADIUS Message Header

Field Name	Semantics	Field Length
Code	Accounting-Request = 4 Accounting-Response = 5	1 byte
Identifier	Used to match accounting-request and accounting-response messages.	1 byte
Length	Total length of RADIUS message min value = 20 max value = 4096	2 bytes
Authenticator	Computed as per RADIUS Specification	16 bytes

Two standard RADIUS attributes MUST follow the RADIUS Message Header: NAS-IP-Address and Acct_Status_Type. These two fields are included to improve interoperability with existing RADIUS server implementations since they are mandatory attributes in a RADIUS Accounting-Request packet.

The NAS-IP-Address indicates the originator of the Accounting-Request message and MUST contain the IP address of the originating PacketCable network element.

The Acct-Status-Type attribute typically indicates whether the Accounting-Request marks the beginning of the user service (Start) or the end (Stop). A PacketCable Accounting-Request message may contain the beginning, end, or update of the user service. For this reason, an Acct-Status-Type value of Interim-Update is used to represent PacketCable Event Messages.

Table 23 - Mandatory RADIUS Attributes

Name	Type	Length	Value
NAS-IP-Address	4	6	IP address of originating PacketCable network element
Acct-Status-Type	40	6	Interim-Update = 3

Table 24 - RADIUS Acct_Status_Type

Type	Length	Value
40	6 bytes	Interim-Update = 3

PacketCable attributes are encoded in the RADIUS Vendor Specific Attributes (VSA) structure as described in this section. Additional PacketCable or vendor-specific attributes can be added to existing Event Messages by adding additional RADIUS VSAs to the message.

The Vendor-Specific attribute includes a field to identify the vendor, and the Internet Assigned Numbers Authority (IANA) has assigned PacketCable an SMI Network Management Private Enterprise Number of 4491 for the encoding of these attributes.

Table 25 - Radius VSA Structure for PacketCable Attributes

Field Name	Semantics	Field Length
Type	Vendor Specific = 26	1 byte
Length	Total Attribute Length Note: Value is Vendor Length + 8	1 byte
Vendor ID	CableLabs = 4491	4 bytes
Vendor Attribute Type	PacketCable Attribute Type	1 byte (see Table 16)

Field Name	Semantics	Field Length
Vendor Attribute Length	PacketCable Attribute Length	1 byte (see Table 16) Note: Value is Vendor Length +2
Vendor Attribute Value	PacketCable Attribute Value	Vendor Length bytes

7.6.3 PacketCable RADIUS Accounting-Request Packet Syntax

```
<<RADIUS Accounting-Request> ::==
                <RADIUS message Header>
                <RADIUS NAS-IP-Address Attribute>
                <RADIUS Acct-Status-Type Attribute>
                <Packet Cable EM>

<Packet Cable EM> ::==
                <RADIUS VSA for PacketCable EM Header Attribute>
                <PacketCable EM Attribute List>

<PacketCable EM Attribute List> ::==
                <RADIUS VSA for PacketCable EM Attribute> |
                <PacketCable EM Attribute List>
                <RADIUS VSA for Packet Cable EM Attribute>
```

The Event Message Header is the first attribute within a given Event Message. The order of the Event Message attributes which follow the Event Message Header is arbitrary.

8   SECURITY REQUIREMENTS

Security for PacketCable Multimedia interfaces utilizes security mechanisms defined in [PKT-SEC1.5] as well as in [DOCSIS MULPIv3.0]. The following table provides a summary of security mechanisms for each of the PacketCable Multimedia interfaces.

Table 26 - Multimedia Security Interfaces

Interface	Description	Security Mechanisms
pkt-mm-1	CMTS – CM	HMAC-based authentication defined by the DOCSIS specification 0.
pkt-mm-2	PS – CMTS	IPsec ESP using IKE or Kerberos-based key management.
pkt-mm-3	AM – PS	IPsec ESP using IKE or Kerberos-based key management.
pkt-mm-4	PS – RKS	IPsec ESP using IKE or Kerberos-based key management.
pkt-mm-5	CMTS – RKS	IPsec ESP using IKE or Kerberos-based key management.
pkt-mm-6	Client – CMTS	Out of scope for this version of this specification.
pkt-mm-7	Client – AM	Out of scope for this version of this specification.
pkt-mm-8	AM – Peer	Out of scope for this version of this specification.
pkt-mm-9	CMTS – MSO-Managed IP Network	Out of scope for this version of this specification.
pkt-mm-10	Client – Peer	Out of scope for this version of this specification.

The following subsections describe security that is applied to each PacketCable Multimedia interface and specify additional requirements or extensions whenever necessary.

8.1   CMTS–CM QoS Interface (pkt-mm-1)

DOCSIS QoS messages are authenticated using an HMAC (Hash Message Authentication Code), which is a keyed cryptographic hash. Calculation of the HMAC attribute that must be included in DOCSIS QoS messages is specified in section C.1.4.1 of [DOCSIS MULPIv3.0].

8.2   Policy Server–CMTS COPS Interface (pkt-mm-2)

The Policy Server–CMTS COPS interface MUST be secured using the IPsec ESP protocol, as specified in section 7.2.1.3.2 of [PKT-SEC1.5]. The key management requirements for this interface MUST comply with section 7.2.1.4.1 of [PKT-SEC1.5]. For this interface, Policy Server MUST comply with all the Gate Controller requirements listed in sections 7.2.1.3.2 and 7.2.1.4.1 of [PKT-SEC1.5]. IKE with pre-shared keys is required to implement, while IKE with certificates and Kerberized IPsec are both optional to implement.

In the case that Kerberized IPsec is utilized, section 6.4.5 of [PKT-SEC1.5] defines principal names of various Kerberized services. The first component of a principal name is unique for each type of a Kerberized service. section 6.4.5 of [PKT-SEC1.5] already specifies the first component of the CMTS's principal name. The first component of Policy Server's principal name MUST be:

<p style="text-align:center">policyserver:<ElementID></p>

where <ElementID> is defined in section 6.4.5 of [PKT-SEC1.5].

In the case that IKE with certificates is utilized, the subject name in a server certificate has the following attribute defined in section 8.2.3.4.3 of [PKT-SEC1.5]:

<p style="text-align:center">OU=<Sub-System Name></p>

The value of <Sub-System Name> identifies a server type. The value of <Sub-System Name> for a CMTS is already specified in section 8.2.3.4.3 of [PKT-SEC1.5]. The value of <Sub-System Name> for a Policy Server MUST be the following string: policyserver.

8.3   Application Manager–Policy Server COPS Interface (pkt-mm-3)

The Application Manager–Policy Server COPS interface MUST be secured using the IPsec ESP protocol, as specified in section 7.2.1.3.2 of [PKT-SEC1.5]. The key management requirements for this interface MUST comply with section 7.2.1.4.1 of [PKT-SEC1.5]. For this interface, Application Manager MUST comply with all the Gate Controller requirements listed in Sections 7.2.1.3.2 and 7.2.1.4.1 of [PKT-SEC1.5]. IKE with pre-shared keys is required to implement, while IKE with certificates and Kerberized IPsec are both optional to implement.

In the case that Kerberized IPsec is utilized, section 6.4.5 of [PKT-SEC1.5] defines principal names of various Kerberized services. The first component of a principal name is unique for each type of a Kerberized service. The first component of Policy Server's principal name is specified in Section 8.2 of this document. The first component of Application Manager's principal name MUST be:

am:<ElementID>

where <ElementID> is defined in section 6.4.5 of [PKT-SEC1.5].

In the case that IKE with certificates is utilized, the subject name in a server certificate has the following attribute defined in section 8.2.3.4.3 of [PKT-SEC1.5]:

OU=<Sub-System Name>

The value of <Sub-System Name> identifies a server type. The value of <Sub-System Name> for a Policy Server is specified in Section 8.2 of this document. The value of <Sub-System Name> for an Application Manager MUST be the following 2-character string: am.

8.4   Policy Server–RKS Event Message Interface (pkt-mm-4)

The Policy Server–RKS Event Message interface MUST be secured using the IPsec ESP protocol, as specified in section 7.3.2 of [PKT-SEC1.5]. The key management for this interface MUST be identical to the one specified for a CMTS-RKS interface in section 7.3.3.2 of [PKT-SEC1.5]. IKE with pre-shared keys is required to implement, while IKE with certificates and Kerberized IPsec are both optional to implement.

In the case that Kerberized IPsec is utilized, section 6.4.5 of [PKT-SEC1.5] defines principal names of various Kerberized services. The first component of a principal name is unique for each type of a Kerberized service. Section 6.4.5 of [PKT-SEC1.5] already specifies the first component of the RKS's principal name. The first component of Policy Server's principal name is specified in Section 8.2 of this document.

In the case that IKE with certificates is utilized, the subject name in a server certificate has the following attribute defined in section 8.2.3.4.3 of [PKT-SEC1.5]:

OU=<Sub-System Name>

The value of <Sub-System Name> identifies a server type. The value of <Sub-System Name> for an RKS is already specified in section 8.2.3.4.3 of [PKT-SEC1.5]. The value of <Sub-System Name> for a Policy Server is specified in section 8.2 of this document.

8.5   CMTS–RKS Event Message Interface (pkt-mm-5)

The CMTS–RKS Event Message interface MUST be secured using the IPsec ESP protocol, as specified in section 7.3.2 of [PKT-SEC1.5]. The key management for this interface is specified in section 7.3.3.2 of [PKT-SEC1.5]. IKE with pre-shared keys is required to implement, while IKE with certificates and Kerberized IPsec are both optional to implement.

9   MAPPING A FLOWSPEC TRAFFIC PROFILE TO DOCSIS

A Traffic Profile defines the QoS attributes of the IP flow or the DOCSIS Service Flow to be used in performing authorization, reservation and commit operations. A Traffic Profile can be defined via one of the following methods:

- FlowSpec

- DOCSIS Service Class Name

- DOCSIS Specific Parameterization

- Upstream Drop

This section describes the mapping procedures for deriving the DOCSIS-specific QoS parameters from the various Traffic Profile representations, excluding Upstream Drop. A Traffic Profile may include the authorization, reservation or commit envelopes. As defined in [IETF RFC 2210], a FlowSpec consists of a TSpec and an optional RSpec.

9.1   Mapping FlowSpecs to DOCSIS Scheduling Types

FlowSpecs support two types of services: Controlled Load and Guaranteed. Controlled Load services provide minimum bandwidth guarantees, but not latency/delay guarantees. Guaranteed services provide both bandwidth and latency/delay guarantees. Guaranteed service may be closely approximated through DOCSIS Real-Time Polling and UGS scheduling types. Controlled Load service may be closely approximated through the DOCSIS Best-Effort scheduling type. The FlowSpec service number in the FlowSpec definition distinguishes between Controlled Load and Guaranteed services. Service number 5 indicates the definition is for Controlled Load service, and service number 2 indicates the definition is for Guaranteed service. Further, Controlled Load service contains only the TSpec token bucket parameters, but not the RSpec. Guaranteed service MUST contain both the TSpec and the RSpec.

For latency and jitter-sensitive applications such as voice, MPEG video or gaming, one could request Guaranteed service. The CMTS can then use the traffic profile parameters specified in the FlowSpec to select one of the two types of DOCSIS scheduling types that could provide Guaranteed service: RTPS and UGS. For other applications that are not latency sensitive one could request Controlled Load service, which can be used to provide minimum bandwidth guarantees. Table 27 below summarizes the choices.

Table 27 - Mapping FlowSpecs Types

DOCSIS Scheduling Type	FlowSpec Service Number	Application Example
Unsolicited Grant Service (UGS)	2 (Guaranteed)	Voice over IP
Real-Time Polling Service (RTPS)	2 (Guaranteed)	VPN
Best Effort (BE)	5 (Controlled Load)	Best Effort Internet Data

The general FlowSpec-to-DOCSIS mapping procedure for upstream service flows is as follows:

- Upon receipt of a Gate-Set message with a FlowSpec the CMTS MUST analyzes the TSpec Service Header to determine whether Controlled load or Guaranteed service is being requested.

- If Controlled Load Service, then the CMTS MUST use only the TSpec parameters to resolve the DOCSIS scheduling parameters to define the DOCSIS Traffic Parameters for a DOCSIS Best-Effort Scheduling type.

- If Guaranteed Service, the CMTS MUST examine the TSpec and RSpec parameters and use either UGS or RTPS scheduling types based on their definitions in Sections 9.3.1 and 9.3.2 respectively.

- If the Reserved Rate (R) and Bucket Rate (r) are not equal, then the CMTS MUST use the TSpec and the RSpec to define the DOCSIS Traffic Parameters for a DOCSIS Real-Time Polling scheduling type.

Note that two other types of DOCSIS scheduling types are not mentioned above. These are:

- Unsolicited Grant Service with Activity Detection
- Non-Real-Time Polling Service

If the Application Manager wishes to request either one of these services, it can only do so by using either the Service Class Name or the DOCSIS-specific parameterization method of defining the Traffic Profile.

9.2   Mapping FlowSpecs to DOCSIS Traffic Parameters

The FlowSpec is made up of two parts, the TSpec and the RSpec. The TSpec describes the traffic for the flow, and the RSpec describes the desired service; note that for a controlled load service, the RSpec is not used. The RSpec parameters MUST be specified for a guaranteed service. The CMTS MUST ignore the RSpec parameters for a controlled load service. The RSpec is used to provide latency guarantees for guaranteed services. Please refer to RFCs 2210 [IETF RFC 2210], 2211 [IETF RFC 2211], and 2212 [IETF RFC 2212] for more information on how these parameters should be used by Application Managers to specify the traffic profile. Note that the PacketCable Multimedia interpretation of FlowSpecs differs from the RFCs in the following respects:

- Guaranteed Service as defined in [IETF RFC 2212] controls Layer 3 queuing delay (i.e., the delays associated with packet scheduling), whereas in PacketCable Multimedia we are primarily concerned with controlling the access delay of the DOCSIS MAC layer. Consequently we reserve bandwidth resources according to the Tspec's r parameter rather than the Rspec's R.

- As defined in [IETF RFC 2211], Controlled Load service defines only a guaranteed minimum rate for a flow. PacketCable Multimedia's Controlled Load service facilitates definition of the maximum rate for a flow, as well as definition of flows without a guaranteed minimum rate.

- The Guaranteed Service Slack Term parameter is not need in PacketCable Multimedia, so the field is redefined to enable control of DOCSIS polling jitter.

TSpec Parameters:

- Bucket Depth (b), bytes
- Bucket Rate (r), bytes/second
- Maximum Datagram Size (M), bytes
- Minimum Policed Unit (m), bytes
- Peak Rate (p), bytes/second

RSpec Parameters:

- Reserved Rate (R), bytes/second
- Slack Term (S), microseconds

The parameter mapping, roughly approximated, involves the following associations for DOCSIS upstream BE (Best-Effort) and downstream Controlled Load Service Flows. The actual mapping procedure would involve normalizing these parameters to account for Layer 2 and Layer 3 header considerations.

- TSpec Bucket Depth (b) ≈ DOCSIS Maximum Traffic Burst
- TSpec Maximum Datagram Size (M) ≈ <not required by DOCSIS >
- TSpec Minimum Policed Unit (m) ≈ DOCSIS Assumed Minimum Reserved Rate Packet Size
- TSpec Bucket Rate (r) ≈ DOCSIS Minimum Reserved Rate
- TSpec Peak Rate (p) ≈ DOCSIS Maximum Sustained Rate and, for DOCSIS 3.0 only, DOCSIS Downstream Peak Traffic Rate

For downstream Guaranteed service flows, the RSpec parameters are added to provide latency and reservation guarantees.

- TSpec Bucket Depth (b) ≈ DOCSIS Maximum Traffic Burst

- TSpec Maximum Datagram Size (M) ≈ <not required by DOCSIS >

- TSpec Minimum Policed Unit (m) ≈ DOCSIS Assumed Minimum Reserved Rate Packet Size

- TSpec Bucket Rate (r) ≈ DOCSIS Minimum Reserved Rate and DOCSIS Maximum Sustained Rate

- TSpec Peak Rate (p) ≈ For DOCSIS 3.0 only, DOCSIS Downstream Peak Traffic Rate

- RSpec Reserved Rate (R) ≈ <not required by DOCSIS>

- RSpec Slack Term ≈ DOCSIS Downstream Latency

The parameter mapping, roughly approximated, involves the following associations for DOCSIS UGS Service Flows.

- TSpec Bucket Depth (b) = TSpec Maximum Datagram Size (M) = TSpec Minimum Policed Unit (m) ≈ DOCSIS Unsolicited Grant Size

- TSpec Bucket Rate (r) = TSpec Peak Rate (p) = RSpec Reserved Rate (R) ≈ used to calculate Nominal Grant Interval

- RSpec Slack Term ≈ DOCSIS Tolerated Grant Jitter

Similarly, the following associations apply for DOCSIS Real-Time Polling Service Flows.

- TSpec Bucket Depth (b) ≈ DOCSIS Maximum Traffic Burst

- TSpec Maximum Datagram Size (M) ≈ <not required by DOCSIS >

- TSpec Bucket Rate (r) ≈ DOCSIS Maximum Sustained Rate and DOCSIS Minimum Reserved Traffic Rate

- RSpec Reserved Rate (R) ≈ used to calculate the Polling Interval

- RSpec Slack Term ≈ Tolerated Polling Jitter

This abstraction model allows standards-based RSVP implementations (as anticipated in Scenarios 2 and 3) to request and receive controlled load or guaranteed service from the network without necessarily requiring DOCSIS-specific info.

In some situations, where the Application Manager and the Policy Server is acutely aware of DOCSIS, it MAY specify the Traffic Profile for the Gate using the DOCSIS Service Class Name or the DOCSIS-specific parameterization format.

Note that there are several DOCSIS Service Flow parameters that cannot be directly resolved from the FlowSpecs; in these cases, the PacketCable Multimedia specification defines defaults for those Service Flow parameters. If the Application Manager/Policy Server wishes to set those Service Flow parameters to something other than the defaults specified by this specification, the Application Manager/Policy Server MUST use either the Service Class Names or the DOCSIS-specific parameterization formats to define the traffic profile.

For Guaranteed Service, the Minimum Reserved Rate and the Maximum Sustained Rates are set to the same value, and are based on the Bucket Rate, 'r'. This is because Guaranteed Service provides latency guarantees, and this means a flow cannot be sustained at a rate greater than the rate at which the source has agreed to generate (when the reservation was initially made). A reservation made with a Traffic Profile specifying a Bucket Rate 'r' means that the source will not sustain a traffic flow greater than 'r'. Thus, it would be incorrect to use the Reserved Rate 'R' to represent any DOCSIS sustained rate (either minimum or maximum), in the Guaranteed service case.

For real-time polling Scheduling however, the CMTS uses the Reserved Rate R to calculate the polling interval, so that traffic sources can burst at the rate of R without increasing the delay that the packets experience waiting for a DOCSIS upstream transmission opportunity. Although the traffic source may generate traffic at rate 'R' in this case, the CMTS will ensure the sustained rate does not violate 'r' over time.

For Controlled Load Service, because there are no latency guarantees and because we want to enable one to use the DOCSIS-specific concepts of guaranteed minimum as well as maximum sustained rates, the TSpec Bucket Rate 'r' is mapped to the DOCSIS minimum rate, and the TSpec Peak Rate 'p' is mapped to the DOCSIS Maximum Sustained Rate. If a zero or infinite value is indicated for 'r', then the DOCSIS Minimum Reserved Rate parameter MUST be omitted. If a zero or infinite value is indicated for 'p', then the DOCSIS Maximum Sustained Rate parameter MUST be omitted.

If a syntactic or semantic conflict exists between the DOCSIS RFI specification and this specification the DOCSIS RFI specification MUST take precedence unless otherwise noted.

Note, the PCMM specification now includes a few new additional parameters to the DOCSIS traffic profiles, to match the service flow parameters in [DOCSIS MULPIv3.1]. There are no new use cases driving the need to map FlowSpec parameters to those new traffic profile parameters at this time.

9.3   DOCSIS Upstream Parameters

For all of the upstream packet size calculations, the following formula must be used unless otherwise specified: The packet PDU MUST be calculated from the first byte following the DOCSIS MAC HCS to the end of the CRC. This value includes the Ethernet header overhead of 18 bytes (6 bytes for source address, 6 bytes for destination address, 2 bytes for length, and 4 bytes for CRC). The value also incorporates DOCSIS MAC layer overhead, including the DOCSIS base header (6 bytes), the UGS extended header (3 bytes), and the BPI+ extended header (5 bytes).

In the equations used in all subsequent sections, the following variables apply:

ENET = Ethernet overhead (18 or 22-bytes), a default of 18-bytes MUST be used unless otherwise indicated (how the CMTS determines to use 22-bytes is outside the scope of Multimedia Specification). In the case of a UGS flow, packets using extended Ethernet headers that are not accounted for will result in the packets being dropped (packets exceeding the Grant size must be dropped). In the case of a RTPS flow, packets using extended Ethernet headers that are not accounted for will result in the packets being sent on the primary (best effort) service flow.

DOCSIS = DOCSIS header = 6 bytes

BPI = DOCSIS BPI header = 5 bytes

UGS = DOCSIS UGS extended header = 3 bytes

9.3.1   Unsolicited Grant Scheduling (UGS)

Unsolicited Grant Scheduling MUST be used when the Service Number is 2 (Guaranteed), the Peak Rate, Bucket Rate, and the Reserved Rate are all equal, and Maximum Datagram Size is equal to Minimum Policed Unit.

The DOCSIS upstream objects MUST be set as stated below. All service flow quality of service TLV encodings that are not defined here MUST be given their default values as indicated by DOCSIS.

The DOCSIS Unsolicited Grant Size incorporates DOCSIS MAC layer overhead in addition to the packet PDU size calculated using the formula specified in section 9.3. The DOCSIS MAC layer overhead includes the DOCSIS base header (6 bytes), the UGS extended header (3 bytes), and optionally the BPI+ extended header (5 bytes).

$$DOCSIS\ Unsolicited\ Grant\ Size = M + ENET + DOCSIS + UGS + BPI$$

The example above assumes that BPI+ [BPI+] is enabled.

The DOCSIS Maximum Sustained Traffic Rate and DOCSIS Assumed Minimum Reserved Rate Packet Size parameters MUST NOT be used for upstream flows.

The DOCSIS Grants per Interval parameter MUST be set to 1.

The DOCSIS Nominal Grant Interval parameter MUST be set to the Maximum Datagram Size divided by the Reserved Rate converted to microseconds.

$$DOCSIS\ Nominal\ Grant\ Interval = M/R * 1,000,000$$

The DOCSIS Tolerated Grant Jitter parameter MUST be set to Slack Term. The minimum allowed value is 800 μs. If the CMTS receives a Gate-Set message with a Slack term less than 800 μs, the CMTS MUST reply with a Gate-Set-Err message with a PacketCable Error-Code of "Invalid Field Value in Object".

The DOCSIS Nominal Polling Interval parameter MUST NOT be specified in the Traffic Profile for UGS service flows. The DOCSIS Tolerated Polling Jitter parameter MUST NOT be specified in the Traffic Profile for UGS service flows. The DOCSIS Request/Transmission Policy parameter is a bitmask; bits 0-6 and 8 MUST be set for UGS service flows. Bit 9 in the Request/Transmission Policy enables/disables the use of segment headers in DOCSIS 3.0 service flows. Bit 9 MUST be set for DOCSIS 3.0 UGS service flows to disable the use of segment headers.

9.3.2   Real-Time Polling Scheduling

Real-Time Polling Scheduling MUST be used when the Service Number is 2 (Guaranteed Service) and either the Peak Rate, Bucket Rate and Reserved Rate are not all equal (i.e., !(p==r==R)), or the Maximum Datagram Size is not equal to the Minimum Policed Unit.

The DOCSIS upstream objects MUST be set as stated below. All service flow quality of service TLV encodings that are not defined here MUST be given their default values as indicated by DOCSIS.

The DOCSIS Maximum Sustained Traffic Rate parameter is given in bits per second, and includes Ethernet layer overhead. The conversion from IP-specific parameters involves first determining the packetization rate by dividing the Bucket Rate by the Minimum Policed Unit. This value is then multiplied by the packet size, Minimum Policed Unit, including MAC layer overhead, and the entire product is scaled from bytes to bits.

$$\text{DOCSIS Maximum Sustained Traffic Rate} = r/m * (m + ENET) * 8$$

The DOCSIS Maximum Traffic Burst parameter MUST be set to the greater of: (1) Bucket Depth including the Ethernet overhead calculated using the Minimum Policed Unit or (2) the DOCSIS specified minimum value of 1522.

$$\text{DOCSIS Maximum Traffic Burst} = max \left((\text{Bucket Depth} / m) * (m + ENET) , 1522 \right)$$

The DOCSIS Minimum Reserved Traffic Rate parameter is the same as the DOCSIS Maximum Sustained Traffic Rate.

$$\text{DOCSIS Minimum Reserved Traffic Rate} = r/m * (m + ENET) * 8$$

The DOCSIS Request/Transmission Policy parameter is a bitmask; the recommended default value should be 0x1F.

The DOCSIS Nominal Polling Interval parameter MUST be set to Minimum Policed Unit divided by Reserved Rate, converted to microseconds.

$$\text{DOCSIS Nominal Polling Interval} = m/R * 1,000,000$$

The DOCSIS Tolerated Polling Jitter parameter MUST be set to Slack Term. The minimum non-zero allowed value is 800 μs. If the CMTS receives a Gate-Set message with a Slack Term not equal to zero and less than 800 μs, the CMTS MUST reply with a Gate-Set-Err message with a PacketCable Error-Code of "Invalid Field Value in Object". Upon receipt of a Slack Term value of 0 the CMTS MUST utilize its implementation-specific default size for this parameter, not 0 microseconds.

$$\text{DOCSIS Nominal Polling Jitter} = S$$

9.3.3   Best Effort Scheduling

Best Effort Scheduling MUST be used when the Service Number is 5 (Controlled-Load).

The DOCSIS upstream objects MUST be set as stated below. All service flow quality of service TLV encodings that are not defined here MUST be given their default values as indicated by DOCSIS.

The DOCSIS Traffic Priority MUST be set to 5.

The DOCSIS Maximum Sustained Traffic Rate parameter is given in bits per second, including Ethernet layer overhead. The conversion from IP-specific parameters involves first determining the packetization rate by dividing the Peak Rate by the Minimum Policed Unit. This value is then multiplied by the packet size, Minimum Policed

Unit, amended to include Ethernet layer overhead, and the entire product is scaled from bytes to bits. The DOCSIS Maximum Sustained Traffic Rate MUST be converted from the Minimum Policed Unit.

$$\text{DOCSIS Maximum Sustained Traffic Rate} = p/m * (m + ENET) * 8$$

The DOCSIS Maximum Traffic Burst parameter MUST be set to the greater of: (1) Bucket Depth including the Ethernet overhead calculated using the Minimum Policed Unit or (2) the DOCSIS specified minimum value of 1522.

$$\text{DOCSIS Maximum Traffic Burst} = \max ((\text{Bucket Depth} / m) * (m + ENET) , 1522)$$

The DOCSIS Minimum Reserved Traffic Rate parameter is calculated in a manner similar to the DOCSIS Maximum Sustained Traffic Rate, except that instead of using the Peak Rate parameter, the Bucket Rate is used.

$$\text{DOCSIS Minimum Reserved Traffic Rate} = r/m * (m + ENET) * 8$$

9.3.4   Upstream Packet Classification Encodings

9.3.4.1   DOCSIS Upstream Packet Classification Requests

The DOCSIS upstream classification objects MUST be set as stated below. All classification TLV encodings that are not defined here MUST be given their default values as indicated by DOCSIS.

The DOCSIS Classifier Identifier parameter MUST be used. If the Extended Classifier is used, a CMTS may choose to use the ClassifierID specified by the AM, however, there are certain cases where this may not work, such as when using MGPI.

The DOCSIS Service Flow Identifier parameter MUST be used.

The DOCSIS Rule Priority parameter MUST be set to the Priority value specified in the classifier object.

The DOCSIS Classification Activation State parameter MUST be set as follows:

- When the Classifier object is used, it MUST be set to active (1) when the Gate utilizing the service flow is committed, and for all the other cases it MUST be set to inactive (0).

- When the Extended Classifier object is used, it MUST be set to the value specified in the Activation State field.

The DOCSIS Dynamic Service Change Action MUST be set as follows:

- When the Classifier object is used, the CMTS MAY use the DSC Add Classifier (0), DSC Replace Classifier (1) and DSC Delete Classifier (2) operations per the DOCSIS RFI specification.

- When the Extended Classifier object is used, it MUST be set to the value specified in the Action field.

The DOCSIS IP Protocol parameter MUST be set to the Protocol ID value specified in the classifier object if the value is non-zero, and omitted otherwise.

The DOCSIS IP Source Address parameter MUST be set to the source address provided in the classifier object, so long as a non-zero value is provided. If the address specified in the classifier object is zero, this parameter MUST be omitted.

The DOCSIS IP Source Mask parameter MUST be set as follows:

- When the Classifier object is used, the DOCSIS IP Source Mask parameter MUST be omitted.

- When the Extended Classifier object is used, the DOCSIS IP Source Mask parameter MUST be set to the same value as the IP Source Mask field.

In the case of the Classifier object, the DOCSIS IP Source Port Start and DOCSIS IP Source Port End parameters MUST be set to the same port value as the Classifier object, so long as a non-zero value is provided. If the value specified in the Classifier object is zero, both the DOCSIS IP Source Port Start and DOCSIS IP Source Port End parameters MUST be omitted.

In the case of the Extended Classifier object, the DOCSIS IP Source Port Start and DOCSIS IP Source Port End parameters MUST be set to the corresponding source port values defined in the Extended Classifier object unless

wildcarded ports are being used. That is, if Source Port Start = 0 and Source Port End = 65535 in the Extended Classifier, these DOCSIS parameters MAY be omitted.

The DOCSIS IP Destination Address parameter MUST be set to the destination address provided in the classifier object, so long as a non-zero value is provided. If the address specified in the classifier object is zero, this parameter MUST be omitted.

The DOCSIS IP Destination Mask parameter MUST be set as follows:

- When the Classifier object is used, the DOCSIS IP Destination Mask parameter MUST be omitted.

- When the Extended Classifier object is used, the DOCSIS IP Destination Mask parameter MUST be set to the same value as the IP Destination Mask field.

In the case of the Classifier object, the DOCSIS IP Destination Port Start and DOCSIS IP Destination Port End parameters MUST be set to the same port value as the Classifier object, so long as a non-zero value is provided. If the value specified in the Classifier object is zero, both the DOCSIS IP Destination Port Start and DOCSIS IP Destination Port End parameters MUST be omitted.

In the case of the Extended Classifier object, the DOCSIS IP Destination Port Start and DOCSIS IP Destination Port End parameters MUST be set to the corresponding destination port values defined in the Extended Classifier object unless wildcarded ports are being used. That is, if Destination Port Start = 0 and Destination Port End = 65535 in the Extended Classifier, these DOCSIS parameters MAY be omitted.

The DOCSIS Ethernet LLC Packet Classification Encodings MUST be omitted.

The DOCSIS 802.1P/Q Packet Classification Encodings MUST be omitted.

In the case of the IPv6 Classifier object, the CMTS SHOULD validate the corresponding cable modem referenced by the Gate-Set supports IPv6 as indicated by the modem capabilities TLV 5.39 "IPv6 Support". If only IPv6 classifiers are present in the Gate-Set and the modem does NOT indicate IPv6 support, a Gate-Set-Err SHOULD be returned. If both IPv6 and Extended classifiers are present in the Gate-Set and the modem does NOT indicate IPv6 support, the CMTS MAY proceed with the IPv4 classifier mapping and gate establishment/modification while silently discarding the IPv6 classifier mapping aspect.

9.4   DOCSIS Downstream Parameters

9.4.1   Downstream QoS Encodings for Guaranteed Service

The DOCSIS downstream service flow quality-of-service TLV encodings MUST be set as stated below. All service flow quality of service TLV encodings that are not defined here MUST be given their default values as indicated by DOCSIS.

The downstream DOCSIS parameters are calculated using the DOCSIS MAC header from the byte following the HCS to the end of the CRC. This value includes the Ethernet header overhead.

Based on this overhead, the DOCSIS Assumed Minimum Reserved Rate Packet Size parameter MUST be calculated as:

$$\text{DOCSIS Assumed Minimum Reserved Rate Packet Size} = m + ENET$$

The DOCSIS Maximum Sustained Traffic Rate parameter is given in bits per second, including MAC layer overhead. The conversion from IP-specific parameters involves first determining the packetization rate by dividing the Bucket Rate by the Minimum Policed Unit. This value is then multiplied by the packet size, Minimum Policed Unit, amended to include MAC layer overhead, and the entire product is scaled from bytes to bits. The DOCSIS Maximum Sustained Traffic Rate MUST be calculated as:

$$\text{DOCSIS Maximum Sustained Traffic Rate} = r/m * (m + ENET) * 8$$

The DOCSIS Minimum Reserved Traffic Rate is equal to the DOCSIS Maximum Sustained Traffic Rate.

Note that the DOCSIS Maximum Sustained Traffic Rate and the DOCSIS Minimum Reserved Traffic Rate are calculated slightly differently in PacketCable Multimedia and PacketCable DQoS. PacketCable Multimedia is based

on r and PacketCable DQoS is based on p. This is due to the fact that in DQoS r = p, while in Multimedia these values differ (in which case r is the proper rate value to use).

For CMs which are provisioned in DOCSIS 3.0 mode, a CMTS which enforces the DOCSIS Downstream Peak Traffic Rate parameter MUST supply the DOCSIS Downstream Peak Traffic Rate parameter. The DOCSIS Downstream Peak Traffic Rate parameter MUST be calculated as:

$$\text{DOCSIS Downstream Peak Traffic Rate} = p/m * (m + ENET) * 8$$

If the CMTS does not enforce the DOCSIS Downstream Peak Traffic Rate parameter, it MUST NOT supply the DOCSIS Downstream Peak Traffic Rate parameter.

The DOCSIS Maximum Traffic Burst parameter MUST be set to the greater of: (1) Bucket Depth including the DOCSIS overhead calculated using the Minimum Policed Unit or (2) the DOCSIS specified minimum value of 1522.

$$\text{DOCSIS Maximum Traffic Burst} = \max ((\text{Bucket Depth} / m) * (m + ENET) , 1522)$$

The DOCSIS Traffic Priority parameter MUST be set to 5.

The DOCSIS Downstream Latency parameter MUST be set to Slack Term, if Slack Term is non-zero. If Slack Term is zero, this parameter MUST NOT be populated.

9.4.2 Downstream QoS Encodings for Controlled Load Service

The DOCSIS downstream service flow quality of service TLV encodings MUST be set as stated below. All service flow quality of service TLV encodings that are not defined here MUST be given their default values as indicated by DOCSIS.

The downstream DOCSIS parameters are calculated using the DOCSIS MAC header from the byte following the HCS to the end of the CRC. This value includes the Ethernet header overhead.

Based on this overhead, the DOCSIS Assumed Minimum Reserved Rate Packet Size parameter MUST be calculated as:

$$\text{DOCSIS Assumed Minimum Reserved Rate Packet Size} = m + ENET$$

The DOCSIS Maximum Sustained Traffic Rate parameter is given in bits per second, including MAC layer overhead. The conversion from IP-specific parameters involves first determining the packetization rate by dividing the Peak Rate by the Minimum Policed Unit. This value is then multiplied by the packet size, Minimum Policed Unit, amended to include MAC layer overhead, and the entire product is scaled from bytes to bits. The DOCSIS Maximum Sustained Traffic Rate MUST be calculated as:

$$\text{DOCSIS Maximum Sustained Traffic Rate} = p/m * (m + ENET) * 8$$

For CMs which are provisioned in DOCSIS 3.0 mode, a CMTS which enforces the DOCSIS Downstream Peak Traffic Rate parameter MUST supply the DOCSIS Downstream Peak Traffic Rate parameter. The DOCSIS Downstream Peak Traffic Rate parameter is equal to the DOCSIS Maximum Sustained Traffic Rate.

If the CMTS does not enforce the DOCSIS Downstream Peak Traffic Rate parameter, it MUST NOT supply the DOCSIS Downstream Peak Traffic Rate parameter.

The DOCSIS Minimum Reserved Traffic Rate parameter is calculated in a manner similar to the DOCSIS Maximum Sustained Traffic Rate, except that instead of using the Peak Rate, the Bucket Rate is used.

$$\text{DOCSIS Minimum Reserved Traffic Rate} = r/m * (m + ENET) * 8$$

The DOCSIS Maximum Traffic Burst parameter MUST be set to the greater of: (1) Bucket Depth including the DOCSIS overhead calculated using the Maximum Datagram Size or (2) the DOCSIS specified minimum value of 1522.

$$\text{DOCSIS Maximum Traffic Burst} = \max ((\text{Bucket Depth} / M) * (M + ENET) , 1522)$$

The DOCSIS Traffic Priority parameter MUST be set to 5.

The DOCSIS Downstream Latency parameter MUST NOT be populated.

9.4.3   Downstream Packet Classification Encodings

9.4.3.1   DOCSIS Downstream Packet Classification Requests

The DOCSIS downstream classification objects MUST be set as stated below. All classification TLV encodings that are not defined here MUST be given their default values as indicated by DOCSIS.

The DOCSIS Classifier Identifier parameter MUST be used. If the Extended Classifier is used, a CMTS may choose to use the ClassifierID specified by the AM, however, there are certain cases where this may not work, such as when using MGPI.

The DOCSIS Service Flow Identifier parameter MUST be used.

The DOCSIS Rule Priority parameter MUST be set to the Priority value specified in the classifier object.

The DOCSIS Classification Activation State parameter MUST be set as follows:

- When the Classifier object is used, it MUST be set to active (1) when the Gate utilizing the service flow is committed, and for all the other cases it MUST be set to inactive (0).

- When the Extended Classifier object is used, it MUST be set to the value specified in the Activation State field.

The DOCSIS Dynamic Service Change Action MUST be set as follows:

- When the Classifier object is used, the CMTS MAY use the DSC Add Classifier (0), DSC Replace Classifier (1) and DSC Delete Classifier (2) operations per the DOCSIS RFI specification.

- When the Extended Classifier object is used, it MUST be set to the value specified in the Action field.

The DOCSIS IP Protocol parameter MUST be set to the Protocol ID value specified in the classifier object if the value is non-zero, and omitted otherwise.

The DOCSIS IP Source Address parameter MUST be set to the source address provided in the classifier object, so long as a non-zero value is provided. If the address specified in the classifier object is zero, this parameter MUST be omitted.

The DOCSIS IP Source Mask parameter MUST be set as follows:

- When the Classifier object is used, the DOCSIS IP Source Mask parameter MUST be omitted.

- When the Extended Classifier object is used, the DOCSIS IP Source Mask parameter MUST be set to the same value as the IP Source Mask field.

In the case of the Classifier object, the DOCSIS IP Source Port Start and DOCSIS IP Source Port End parameters MUST be set to the same port value as the Classifier object, so long as a non-zero value is provided. If the value specified in the Classifier object is zero, both the DOCSIS IP Source Port Start and DOCSIS IP Source Port End parameters MUST be omitted.

In the case of the Extended Classifier object, the DOCSIS IP Source Port Start and DOCSIS IP Source Port End parameters MUST be set to the corresponding source port values defined in the Extended Classifier object unless wildcarded ports are being used. That is, if Source Port Start = 0 and Source Port End = 65535 in the Extended Classifier, these DOCSIS parameters MAY be omitted.

The DOCSIS IP Destination Address parameter MUST be set to the destination address provided in the classifier object, so long as a non-zero value is provided. If the address specified in the classifier object is zero, this parameter MUST be omitted.

The DOCSIS IP Destination Mask parameter MUST be set as follows:

- When the Classifier object is used, the DOCSIS IP Destination Mask parameter MUST be omitted.

- When the Extended Classifier object is used, the DOCSIS IP Destination Mask parameter MUST be set to the same value as the IP Destination Mask field.

In the case of the Classifier object, the DOCSIS IP Destination Port Start and DOCSIS IP Destination Port End parameters MUST be set to the same port value as the Classifier object, so long as a non-zero value is provided. If the value specified in the Classifier object is zero, both the DOCSIS IP Destination Port Start and DOCSIS IP Destination Port End parameters MUST be omitted.

In the case of the Extended Classifier object, the DOCSIS IP Destination Port Start and DOCSIS IP Destination Port End parameters MUST be set to the corresponding destination port values defined in the Extended Classifier object unless wildcarded ports are being used. That is if Destination Port Start = 0 and Destination Port End = 65535 in the Extended Classifier, these DOCSIS parameters MAY be omitted.

The DOCSIS Ethernet LLC Packet Classification Encodings MUST be omitted.

The DOCSIS 802.1P/Q Packet Classification Encodings MUST be omitted.

In the case of the IPv6 Classifier object, the CMTS SHOULD validate the corresponding cable modem referenced by the Gate-Set supports IPv6 as indicated by the modem capabilities TLV 5.39 "IPv6 Support". If only IPv6 classifiers are present in the Gate-Set and the modem does NOT indicate IPv6 support a Gate-Set-Err SHOULD be returned. If both IPv6 and Extended classifiers are present in the Gate-Set and the modem does NOT indicate IPv6 support the CMTS MAY proceed with the IPv4 classifier mapping and gate establishment/modification while silently discarding the IPv6 classifier mapping aspect.

10 MESSAGE FLOWS

This section provides two interaction scenarios between the various network elements previously introduced in this specification. The first interaction outlines at a relatively high-level the basic message exchanges that take place within the PacketCable Multimedia framework in order to authorize, reserve and commit access network resources under Scenario 1. The second interaction provides a very detailed breakout of each message and attribute involved in the PacketCable Multimedia QoS and EM interfaces.

10.1 Basic Message Sequence
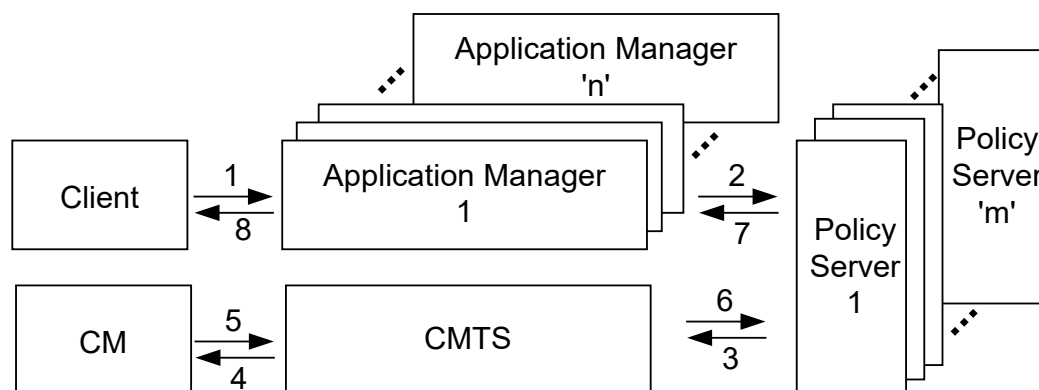
Figure 8 - Basic Message Sequence

1. Client issues a session setup request to the Application Manager via Application Layer signaling. The client may authenticate itself to the Application Manager during this step.

2. Before the Application Manager activates the session, the Application Manager issues a Gate Set (in a COPS DECISION message) and sends it to Policy Server in order to determine if the session setup request should be allowed to proceed. Message includes:

   a. AMID
   b. SubscriberID
   c. TransactionID
   d. Classifier
   e. Traffic Profile for Flow
   f. Gate-Spec

3. Upon receiving the request, the Policy Server checks the request against the policy rules and if the request is approved, sends a Gate-Set to the CMTS. Message includes:

   a. AMID
   b. SubscriberID
   c. TransactionID
   d. Classifier
   e. Traffic Profile for Flow (Authorized, Reserved and Committed)
   f. Gate-Spec

4. If the flow is unshared or if the flow is shared but not yet created, the CMTS uses the classifier and Traffic Profile information to trigger the activation of the flow by issuing the appropriate DOCSIS messages.

5. CM acknowledges with the appropriate DOCSIS messaging.

6. CMTS issues a Gate-Set-Ack to the Policy Server in response to the Gate-Set message received in Step 3. Message includes:

   a. AMID
   b. TransactionID
   c. GateID
   d. SubscriberID
   e. SharedResourceID (if a shared resource is used to fulfill the resource commitment)

7. In response the Policy Server will generate a Gate-Set-Ack to the AM; this tells the AM that the Policy Request has been admitted and the client's request can proceed, and the necessary resources in the underlying network have been reserved. Message includes:

   a. AMID
   b. TransactionID
   c. GateID
   d. SubscriberID
   e. SharedResourceID (if a shared resource is used to fulfill the resource commitment)

8. Application Manager, upon receiving the Gate-Set-Ack will inform the client that the session establishment can proceed.

10.2 Detailed Message Sequence
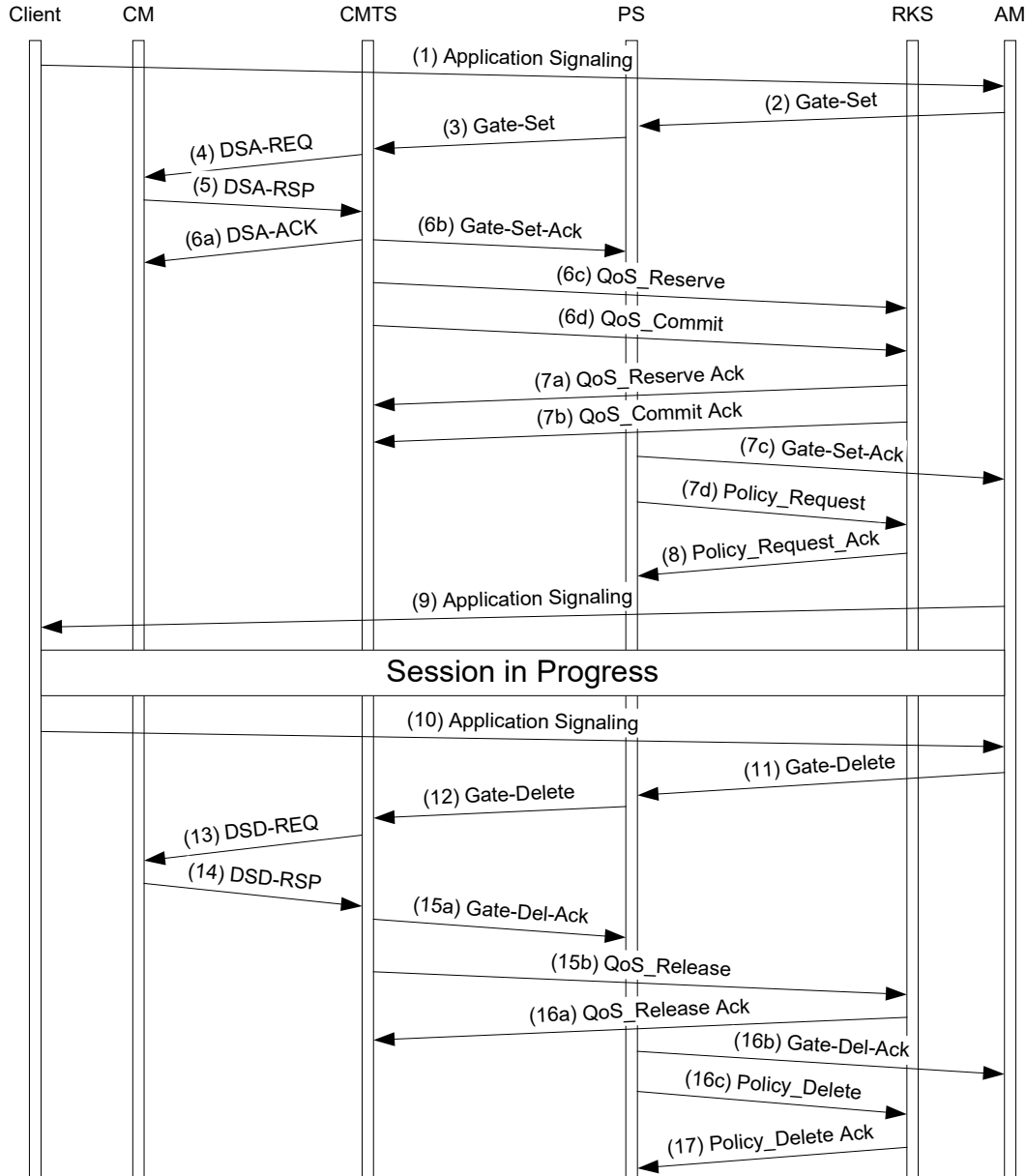
Figure 9 - Detailed Message Sequence

The pages that follow describe in detail the messages that are being exchanged in an example PacketCable Multimedia session. The bandwidth numbers are purely examples, and do not correlate to any particular service. Only the upstream access network resources are being reserved and committed for clarity. Also, BPI related TLVs have been left out of the DOCSIS messages for clarity.

1. The client initializes the session by querying an Application Manager for the necessary resources to use the application. An example of this would be a software based video game, asking for resources to play an online game. This signaling is out of scope for this specification.

2. After receiving the application signaling from the client, the Application Manager issues a Gate-Set to the Policy Server, requesting the needed resources for this session.

0	1	2	3
COPS Header			
Version 0x1	Flags 0x0	Op-Code 0x02	Client-Type 0x800A
Message Length 0x00000088			
COPS Handle Object			
Length 0x0008		C-Num 0x01	C-Type 0x01
Handle 0x00001234			
COPS Context Object			
Length 0x0008		C-Num 0x02	C-Type 0x01
Request Type (R-Type) 0x0008 (Configuration Request)		Message Type (M-Type) 0x0000	
COPS Decision Flags Object			
Length 0x0008		C-Num 0x06	C-Type 0x01
Command Code 0x0001 (Install Configuration)		Flags 0x0000	
COPS Decision Client Specific Decision Data Object Header			
Length 0x00A0		C-Num 0x06	C-Type 0x04
Multimedia TransactionID Object			
Length 0x0008		S-Num 0x01	S-Type 0x01
TransactionID 0x9999		Gate Command 0x0004 (Gate-Set)	
Multimedia AMID Object			
Length 0x0008		S-Num 0x02	S-Type 0x01
Application Type0x0000		Application Manager Tag 0x5678	
Multimedia SubscriberID Object			
Length 0x0008		S-Num 0x03	S-Type 0x01
SubscriberID 0x01010101			
Multimedia GateSpec Object			
Length 0x0010		S-Num 0x05	S-Type 0x01
Flags 0x01	DSCP/TOS Overwrite 0x00	DSCP/TOS Mask 0x00	SessionClassID 0x00
Timer T1 0x00C8 (200 seconds)		Timer T2 0x012C (300 seconds)	

Timer T3	Timer T4
0x003C (60 seconds)	0x001E (30 seconds)

Multimedia FlowSpec Object		
Length	S-Num	S-Type
0x0024	0x07	0x01

Envelope	Service Number	Reserved	Reserved
0x07	0x02		

Combined Authorized, Reserved and Committed Envelopes
Token Bucket Rate [r] (encoded as IEEE floating point)
0x461C4000 (10,000 Bps)
Token Bucket Size [b] (encoded as IEEE floating point)
0x43480000 (200 bytes)
Peak Data Rate [p] (encoded as IEEE floating point)
0x461C4000 (10,000 Bps)
Minimum Policed Unit [m]
0x000000C8 (200 bytes)
Maximum Packet Size [M]
0x000000C8 (200 bytes)
Rate [R] (encoded as IEEE floating point)
0x461C4000 (10,000 Bps)
Slack Term [S]
0x00000320 (800 μs)

Multimedia Classifier Object		
Length	S-Num	S-Type
0x0018	0x06	0x01

Reserved	ProtocolID	DSCP/TOS Field	DSCP/TOS Mask
	0x11 (17 UDP)	0x00	0x00

Source IP Address			
0x01010101			
Destination IP Address			
0x02020202			

Source Port	Destination Port
0x1234	0x9876

Priority	Reserved
0x0040 (64)	

3. After the PS receives the Gate-Set from the Application Manager, it checks to see if the request is authorized, and if so, sends a Gate-Set to the CMTS.

0	1	2	3
COPS Header			

Version	Flags	Op-Code	Client-Type
0x1	0x0	0x02	0x800A

Message Length			
0x000000B4			

COPS Handle Object			

Length	C-Num	C-Type
0x0008	0x01	0x01

Handle 0x00005678		
COPS Context Object		
Length 0x0008	C-Num 0x02	C-Type 0x01
Request Type (R-Type) 0x0008 (Configuration Request)	Message Type (M-Type) 0x0000	
COPS Decision Flags Object		
Length 0x0008	C-Num 0x06	C-Type 0x01
Command Code 0x0001 (Install Configuration)	Flags 0x0000	
COPS Decision Client Specific Decision Data Object Header		
Length 0x00CC	C-Num 0x06	C-Type 0x04
Multimedia TransactionID Object		
Length 0x0008	S-Num 0x01	S-Type 0x01
TransactionID 0x0001	Gate Command 0x0004 (Gate-Set)	
Multimedia AMID Object		
Length 0x0008	S-Num 0x02	S-Type 0x01
AMID 0x0000	Application Manager Tag 0x5678	
Multimedia SubscriberID Object		
Length 0x0008	S-Num 0x03	S-Type 0x01
SubscriberID 0x01010101		
Multimedia GateSpec Object		
Length 0x0010	S-Num 0x05	S-Type 0x01

Direction 0x01	DSCP/TOS Overwrite 0x00	DSCP/TOS Mask 0x00	SessionClassID 0x00
Timer T1 0x00C8 (200 seconds)		Timer T2 0x012C (300seconds)	
Timer T3 0x003C (60 seconds)		Timer T4 0x001E (30 seconds)	
Multimedia FlowSpec Object			
Length 0x0024		S-Num 0x07	S-Type 0x01
Envelope 0x07	Service Number 0x02	Reserved	Reserved
Combined Authorized, Reserved, and Committed Envelopes			

Token Bucket Rate [r] (encoded as IEEE floating point) 0x461C4000 (10,000 Bps)			
Token Bucket Size [b] (encoded as IEEE floating point) 0x43480000 (200 Bytes)			
Peak Data Rate [p] (encoded as IEEE floating point) 0x461C4000 (10,000 Bytes)			
Minimum Policed Unit [m] 0x000000C8 (200 Bytes)			
Maximum Packet Size [M] 0x000000C8 (200 Bytes)			
Rate [R] (encoded as IEEE floating point) 0x461C4000 (10,000 Bytes)			
Slack Term [S] 0x00000320 (800 μs)			
Multimedia Classifier Object			
Length 0x0018		S-Num 0x06	S-Type 0x01
Reserved	ProtocolID 0x11	DSCP/TOS Field 0x00	DSCP/TOS Mask 0x00
Source IP Address 0x01010101			
Destination IP Address 0x02020202			
Source Port 0x1234		Destination Port 0x9876	
Priority 0x0040 (64)		Reserved	
Multimedia Event Generation Info Object			
Length 0x002C		S-Num 0x08	S-Type 0x01
Primary RKS Address 0x03030303			
Primary RKS Port 0x1111		Reserved	
Secondary RKS Address 0x04040404			
Secondary RKS Port 0x1111		Reserved	
BCID 0x3e4812082020202020313436302d3035303030300003db77			

4. If CMTS admission control succeeds, the CMTS will initiate reserving and committing the access network resources by issuing a DSA to the Cable Modem.

0	1	2	3
MAC Management Header			

TransactionID 0x0007		US Service Flow 0x18	Length 0x29
Service Flow ID 0x02	Length 0x04	Value 0x0000	
Value (cont.) 0001		Service ID 0x03	Length 0x02
Value 0x0001		QoS Param Set 0x06	Length 0x01
Value 0x06 (Ad.+Act.)	Scheduling Type 0x0F	Length 0x01	Value 0x06
UGS Size 0x13	Length 0x02	Value 0x00E8 (232 Bytes)	
Nom. Grant Int. 0x14	Length 0x04	Value 0x0000	
Value (cont.) 4E20 (20,000 μs)		Grants Per Interval 0x16	Length 0x01
Value 0x01	RX/TX Policy 0x10	Length 0x04	Value 0x00
Value (cont.) 00037F			Tol. Grant Jitter 0x15
Length 0x04	Value 0x000003		
Value (cont.) 20 (800 μs)	US Pkt. Clfr. 0x16	Length 0x0x2B	Clfr. ID 0x02
Length 0x02	Value 0x0001		Service Flow ID 0x04
Length 0x04	Value 0x000000		
Value (cont.) 01	Rule Priority 0x05	Length 0x01	Value 0x40
Clfr Act. State 0x06	Length 0x01	Value 0x01 (Active)	IP Pkt. Clfr. 0x09
Length 0x001A	IP Protocol 0x02	Length 0x02	Value 0x00
Value (cont.) 11 (17 UDP)	IP Src. Addr 0x03	Length 0x04	Value 0x01
Value (cont.) 010101			IP Src Port Start 0x07
Length 0x02	Value 0x1234		IP Src Port End 0x08
Length 0x02	Value 0x1234		IP Dest Port Start 0x09

| Length 0x02 | Value 0x9876 | IP Dest Port End 0x0A |
| Length 0x02 | Value 0x9876 | |

5. The CM responds to the CMTS with a DSA-RSP.

0	1	2	3
Mac Management Header			
TransactionID 0x0007		Confirm. Code 0x00	

6a. The CMTS completes the transaction with a DSA-ACK

0	1	2	3
Mac Management Header			
TransactionID 0x0007		Confirm. Code 0x00	

6b. Once a DSA-RSP is received by the CMTS from the CM confirming a successful transaction, the CMTS will send a Gate-Set-Ack to the Policy Server.

0	1	2	3
COPS Header			
Version 0x1	Flags 0x1	Op-Code 0x03	Client-Type 0x800A
Message Length 0x0000003C			
COPS Handle Object			
Length 0x0008		C-Num 0x01	C-Type 0x01
Handle 0x00005678			
COPS Report-Type Object			
Length 0x0008		C-Num 0x12	C-Type 0x01
Report Type (R-Type) 0x0001 (Success)		Reserved	
COPS ClientSI Object Header			

Length 0x0024	C-Num 0x09	C-Type 0x01
Multimedia TransactionID Object		
Length 0x0008	S-Num 0x01	S-Type 0x01
TransactionID 0x0001	Gate Command 0x0005 (Gate-Set-Ack)	
Multimedia AMID Object		
Length 0x0008	S-Num 0x02	S-Type 0x01
AMID 0x0000	Application Manager Tag 0x5678	
Multimedia SubscriberID Object		
Length 0x0008	S-Num 0x03	S-Type 0x01
SubscriberID 0x01010101		
Multimedia GateID Object		
Length 0x0008	S-Num 0x04	S-Type 0x01
GateID 0x12345678		

6c. The CMTS will also send a QoS_Reserve Event message to signal to the RKS that the access network resources have been reserved.

0	1	2	3
Accounting-Request Radius Header			
Radius Ven. Spec. 0x1A	Length 0x54	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type (EM Header) 0x01	Length 0x4E
Version 0x0003		BCID 0x3D48	
BCID (cont.) 1208202020202020313436302D3035303030300003DB77			
		Event Message Type 0x0007 (QoS-Reserver)	
Element Type 0x0002 (CMTS)		Element ID 0x2020202031323334	

	Time Zone 0x302D303530303030	
	Sequence Number 0x0000	
Sequence Number (cont.) 0001	Event Time 0x3230	
Event Time (cont.) 3033313230363030303030302E303030		
Status 0x00000000		
Priority 0x80 (128)	Attribute Count 0x0004	Event Object 0x00
Radius Ven. Spec. 0x1A	Length 0x5C	Vendor ID 0x0000
Vendor ID (cont.) 118B	Type 0x20	Length 0x56
QoS_Descriptor 0x0000207D (QoS Status Bitmask – 10000001111101)		
QoS_Descriptor (cont.) 00000000 (Service Class Name)		
QoS_Descriptor (cont.) 00000000 (Service Class Name cont.)		
QoS_Descriptor (cont.) 00000000 (Service Class Name cont.)		
QoS_Descriptor (cont.) 00000000 (Service Class Name cont.)		
QoS_Descriptor (cont.) 00000006 (Service Flow Scheduling Type – UGS)		
QoS_Descriptor (cont.) 00004e20 (Nominal Grant Interval – 20000µs)		
QoS_Descriptor (cont.) 00000320 (Tolerated Grant Jitter – 800 µs)		
QoS_Descriptor (cont.) 00000001 (Grants Per Interval –1)		
QoS_Descriptor (cont.) 000000e8 (Unsolicited Grant Size – 232 bytes)		
QoS_Descriptor (cont.) 0000037F (Request Transmission Policy – 1101111111)		
Radius Ven. Spec. 0x1A	Length 0x0C	Vendor ID 0x0000
Vendor ID (cont.) 118B	Type 0x1E	Length 0x06
SF_ID 0x00000001		

Radius Ven. Spec. 0x1A	Length 0x0A	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type 0x32	Length 0x04
Flow Direction 0x0001 (Upstream)		Radius Ven. Spec. 0x1A	Length 0x0A
Vendor ID 0x0000118B			
Type 0x37	Length 0x04	Element_Requesting_QoS 0x0001 (Policy Server)	

6d. Immediately after sending the QoS_Reserve Event Message to the RKS, the CMTS will send the QoS_Commit Event Message to the RKS. This is due to the fact that the access network resources are being reserved and committed in one step.

0	1	2	3
Accounting-Request Radius Header			
Radius Ven. Spec. 0x1A	Length 0x54	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type (EM Header) 0x01	Length 0x4E
Version 0x0003		BCID 0x3E48	
BCID (cont.) 1208202020202020313436302D3035303030300003DB77			
		Event Message Type 0x0013 (QoS-Commit)	
Element Type 0x0002 (CMTS)		Element ID 0x2020202031323334	
		Time Zone 0x302d303530303030	
		Sequence Number 0x0000	
Sequence Number (cont.) 0002		Event Time 0x3230	
Event Time (cont.) 30333132303630303030303030302E303030			

Status 0x00000000			
Priority 0x80 (128)	Attribute Count 0x0003		Event Object 0x00
Radius Ven. Spec. 0x1A	Length 0x5C	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type 0x20	Length 0x56
QoS_Descriptor 0x0000207D (QoS Status Bitmask – 10000001111101)			
QoS_Descriptor (cont.) 00000000 (Service Class Name)			
QoS_Descriptor (cont.) 00000000 (Service Class Name cont.)			
QoS_Descriptor (cont.) 00000000 (Service Class Name cont.)			
QoS_Descriptor (cont.) 00000000 (Service Class Name cont.)			
QoS_Descriptor (cont.) 00000006 (Service Flow Scheduling Type – UGS)			
QoS_Descriptor (cont.) 00004e20 (Nominal Grant Interval – 20000µs)			
QoS_Descriptor (cont.) 00000320 (Tolerated Grant Jitter – 800 µs)			
QoS_Descriptor (cont.) 00000001 (Grants Per Interval – 1)			
QoS_Descriptor (cont.) 000000e8 (Unsolicited Grant Size – 232 bytes)			
QoS_Descriptor (cont.) 0000037F (Request Transmission Policy – 1101111111)			
Radius Ven. Spec. 0x1A	Length 0x0C	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type 0x1E	Length 0x06
SF_ID 0x00000001			
Radius Ven. Spec. 0x1A	Length 0x0A	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type 0x32	Length 0x04
Flow Direction 0x0001 (Upstream)			

7a. After receiving and recording the QoS_Reserve Event Message, the RKS acknowledges the message.

0	1	2	3
Accounting-Response Radius Header			

7b. After receiving and recording the QoS_Commit Event Message, the RKS acknowledges the message.

0	1	2	3
Accounting-Response Radius Header			

7c. As a result of receiving a Gate-Set-Ack from the CMTS, the Policy Server will send a Gate-Set-Ack to the Application Manager to complete the transaction.

0	1	2	3
COPS Header			
Version 0x1	Flags 0x1	Op-Code 0x03	Client-Type 0x800A
Message Length 0x0000003C			
COPS Handle Object			
Length 0x0008		C-Num 0x01	C-Type 0x01
Handle 0x00001234			
COPS Report-Type Object			
Length 0x0008		C-Num 0x12	C-Type 0x01
Report Type (R-Type) 0x0001 (Success)		Reserved	
COPS ClientSI Object Header			
Length 0x0024		C-Num 0x09	C-Type 0x01
Multimedia TransactionID Object			
Length 0x0008		S-Num 0x01	S-Type 0x01
TransactionID 0x9999		Gate Command 0x0005	
Multimedia AMID Object			
Length 0x0008		S-Num 0x02	S-Type 0x01
AMID 0x0000		Application Manager Tag 0x5678	
Multimedia SubscriberID Object			
Length 0x0008		S-Num 0x03	S-Type 0x01
SubscriberID 0x01010101			
Multimedia GateID Object			

Length		S-Num	S-Type
0x0008		0x04	0x01
GateID			
0x12345678			

7d. The Policy Server will also send a Policy_Request Event Message to the RKS to track the Policy Request and associated outcome.

0	1	2	3
Accounting-Request Radius Header			
Radius Ven. Spec.	Length	Vendor ID	
0x1A	0x54	0x0000	
Vendor ID (cont.)		Type (EM Header)	Length
118B		0x01	0x4E
Version		BCID	
0x0001		0x3E48	
BCID (cont.)			
12082020202020313436302D3035303030300003DB77			
		Event Message Type	
		0x0015 (Policy_Request)	
Element Type		Element ID	
0x0004 (Policy Server)		0x2020202035363738	
		Time Zone	
		0x302E303530303030	
		Sequence Number	
		0x0000	
Sequence Number (cont.)		Event Time	
0001		0x3230	
Event Time (cont.)			
30333132303630303030302E323130			
Status			
0x00000000			
Priority	Attribute Count		Event Object
0x80 (128)	0x0004		0x00

Radius Ven. Spec. 0x1A		Length 0x0C	Vendor ID 0x0000	
Vendor ID (cont.) 118B			Type 0x3D	Length 0x06
Application_Manager_ID 0x00005678				
Radius Ven. Spec. 0x1A		Length 0x0C	Vendor ID 0x0000	
Vendor ID (cont.) 118B			Type 0x34	Length 0x06
Subscriber_ID 0x01010101				
Radius Ven. Spec. 0x1A		Length 0x0A	Vendor ID 0x0000	
Vendor ID (cont.) 118B			Type 0x3C	Length 0x04
Policy_Decision_Status 0x0001 (Policy Approved)			Radius Ven. Spec. 0x1A	Length 0x1C
Vendor ID 0x0000118B				
Type 0x31		Length 0x16	FEID 0x0000	
FEID (cont.) 00000000000000005061636B65744361626C65				

8. After receiving and recording the Policy_Request Event Message, the RKS acknowledges the message.

0	1	2	3
Accounting-Response Radius Header			

9. The Application Manger will reply to the client to inform the client that it can now play the game. This signaling is out of scope for this specification.

10. When the client is finished with the application, it will notify the Application Manager. This signaling is out of scope for this specification.

11. The Application Manager will terminate the session by sending a Gate-Delete to the Policy Server.

0		1	2	3
COPS Header				
Version 0x1	Flags 0x0	Op-Code 0x02	Client-Type 0x800A	

Message Length 0x00000044		
COPS Handle Object		
Length 0x0008	C-Num 0x01	C-Type 0x01
Handle 0x00001234		
COPS Context Object		
Length 0x0008	C-Num 0x02	C-Type 0x01
Request Type (R-Type) 0x0008 (Configuration Request)	Message Type (M-Type) 0x0000	
COPS Decision Flags Object		
Length 0x0008	C-Num 0x06	C-Type 0x01
Command Code 0x0001 (Install Configuration)	Flags 0x0000	
COPS Decision Client Specific Decision Data Object Header		
Length 0x0014	C-Num 0x06	C-Type 0x04
Multimedia TransactionID Object		
Length 0x0008	S-Num 0x01	S-Type 0x01
TransactionID 0x9998	Gate Command 0x000A (Gate-Delete)	
Multimedia AMID Object		
Length 0x0008	S-Num 0x02	S-Type 0x01
AMID 0x0000	Application Manager Tag 0x5678	
Multimedia SubscriberID Object		
Length 0x0008	S-Num 0x03	S-Type 0x01
SubscriberID 0x01010101		
Multimedia GateID Object		
Length 0x0008	S-Num 0x04	S-Type 0x01
GateID 0x12345678		

12. The Policy Server will instruct the CMTS to tear down the session by sending a Gate-Delete

0		1	2	3
COPS Header				
Version 0x1	Flags 0x0	Op-Code 0x02	Client-Type 0x800A	

Message Length		
0x00000044		
COPS Handle Object		
Length	C-Num	C-Type
0x0008	0x01	0x01
Handle		
0x00005678		
COPS Context Object		
Length	C-Num	C-Type
0x0008	0x02	0x01
Request Type (R-Type)	Message Type (M-Type)	
0x0008 (Configuration Request)	0x0000	
COPS Decision Flags Object		
Length	C-Num	C-Type
0x0008	0x06	0x01
Command Code	Flags	
0x0001 (Install Configuration)	0x0000	
COPS Decision Client Specific Decision Data Object Header		
Length	C-Num	C-Type
0x0014	0x09	0x01
Multimedia TransactionID Object		
Length	S-Num	S-Type
0x0008	0x01	0x01
TransactionID	Gate Command	
0x0002	0x000A (Gate-Delete)	
Multimedia AMID Object		
Length	S-Num	S-Type
0x0008	0x02	0x01
AMID	Application Manager Tag	
0x0000	0x5678	
Multimedia SubscriberID Object		
Length	S-Num	S-Type
0x0008	0x03	0x01
SubscriberID		
0x01010101		
Multimedia GateID Object		
Length	S-Num	S-Type
0x0008	0x04	0x01
GateID		
0x12345678		

13. The CMTS will tear down the access network resources by sending a DSD-REQ to the CM.

0	1	2	3
MAC Management Header			

TransactionID 0x0008	Reserved
SFID 0x00000001	

14. The CM will acknowledge the session deletion with a DSD-RSP.

0	1	2	3
MAC Management Header			
TransactionID 0x0008		Confirm. Code 0x00	Reserved

15a. The CMTS will complete the Gate-Control transaction with a Gate-Delete-Ack.

0	1	2	3
COPS Header			
Version 0x1	Flags 0x1	Op-Code 0x03	Client-Type 0x800A
Message Length 0x00000034			
COPS Handle Object			
Length 0x0008		C-Num 0x01	C-Type 0x01
Handle 0x00005678			
COPS Report Type Object			
Length 0x0008		C-Num 0x12	C-Type 0x01
Report Type (R-Type) 0x0001		Reserved	
COPS ClientSI Object Header			
Length 0x001C		C-Num 0x09	C-Type 0x01
Multimedia TransactionID Object			
Length 0x0008		S-Num 0x01	S-Type 0x01
TransactionID 0x0002		Gate Command 0x000B (Gate-Delete-Ack)	
Multimedia AMID Object			
Length 0x0008		S-Num 0x02	S-Type 0x01

AMID 0x0000	Application Manager Tag 0x5678	
Multimedia GateID Object		
Length 0x0008	S-Num 0x04	S-Type 0x01
GateID 0x12345678		

15b. Also upon the receipt of the DSD-RSP, the CMTS will inform the RKS that the network access resources have been freed by sending a QoS_Release.

0	1	2	3
Accounting-Request Radius Header			
Radius Ven. Spec. 0x1A	Length 0x54	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type (EM Header) 0x01	Length 0x4E
Version 0x0001		BCID 0x3E48	
BCID (cont.) 12082020202020313436302D3035303030300003DB77			
		Event Message Type 0x0008 (QoS_Release)	
Element Type 0x0002 (CMTS)		Element ID 0x20202020031323334	
		Time Zone 0x302D303530303030	
		Sequence Number 0x0000	
Sequence Number (cont.) 0003		Event Time 0x3230	
Event Time (cont.) 303231323036303030303030302E333030			
Status 0x00000000			

Priority 0x80 (128)	Attribute Count 0x0005		Event Object 0x00
Radius Ven. Spec. 0x1A	Length 0x0C	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type 0x1E	Length 0x06
SF_ID 0x00000001			
Radius Ven. Spec. 0x1A	Length 0x0A	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type 0x32	Length 0x04
Flow_Direction 0x0001 (Upstream)		Radius Ven. Spec. 0x1A	Length 0x0A
Vendor ID 0x0000118B			
Type 0x38	Length 0x04	QoS_Release_Reason 0x0001 (Gate Closed by PS)	
Radius Ven. Spec. 0x1A	Length 0x0C	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type 0x36	Length 0x06
QoS_Usage_Info 0x77777777 (bytes)			
Radius Ven. Spec. 0x1A	Length 0x0C	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type 0x3F	Length 0x06
QoS_Time_Info 0x77777777 (seconds)			

16a. After receiving and recording the QoS_Release Event Message, the RKS acknowledges the message.

0	1	2	3
Accounting-Response Radius Header			

16b. After receiving the Gate-Delete-Ack from the CMTS, the Policy Server will send a Gate-Delete-Ack to complete the Gate-Control transaction.

0		1	2	3
COPS Header				
Version 0x1	Flags 0x1	Op-Code 0x03	Client-Type 0x800A	
Message Length 0x00000034				
COPS Handle Object				

Length 0x0008	C-Num 0x01	C-Type 0x01
Handle 0x00001234		
COPS Report-Type Object		
Length 0x0008	C-Num 0x12	C-Type 0x01
Report Type (R-Type) 0x0001 (Success)	Reserved	
COPS ClientSI Object Header		
Length 0x001C	C-Num 0x09	C-Type 0x01
Multimedia TransactionID Object		
Length 0x0008	S-Num 0x01	S-Type 0x01
TransactionID 0x9998	Gate Command 0x000B (Gate-Delete-Ack)	
Multimedia AMID Object		
Length 0x0008	S-Num 0x02	S-Type 0x01
AMID 0x0000	Application Manager Tag 0x5678	
Multimedia GateID Object		
Length 0x0008	S-Num 0x04	S-Type 0x01
GateID 0x12345678		

16c. The Policy Server sends a Policy_Delete Event Message to the RKS to complete the entire process.

0	1	2	3
Accounting-Request Radius Header			
Radius Ven. Spec. 0x1A	Length 0x54	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type (EM Header) 0x01	Length 0x4E
Version 0x0001		BCID 0x3E48	
BCID (cont.) 12082020202020313436302D3035303030300003DB77			

	Event Message Type 0x0016 (Policy_Delete)		
Element Type 0x0004 (Policy Server)	Element ID 0x2020202035363738		
	Time Zone 0x302D303530303030		
	Sequence Number 0x0000		
Sequence Number (cont.) 0002	Event Time 0x3230		
Event Time (cont.) 30323132303630303030302E343030			
Status 0x00000000			
Priority 0x80	Attribute Count 0x0004		Event Object 0x00
Radius Ven. Spec. 0x1A	Length 0x0C	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type 0x3D	Length 0x06
Application_Manager_ID 0x00005678			
Radius Ven. Spec. 0x1A	Length 0x0C	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type 0x34	Length 0x06
Subscriber_ID 0x01010101			
Radius Ven. Spec. 0x1A	Length 0x0A	Vendor ID 0x0000	
Vendor ID (cont.) 118B		Type 0x3A	Length 0x04
Policy_Deleted_Reason 0x0001 (Application Manager Request)		Radius Ven. Spec. 0x1A	Length 0x1C
Vendor ID 0x0000118B			
Type 0x31	Length 0x16	FEID 0x0000	
FEID (cont.) 000000000000005061636B65744361626C65			

<table>
<tr><td></td></tr>
</table>

17. After receiving and recording the Policy_Delete Event Message, the RKS acknowledges the message.

0	1	2	3
Accounting-Response Radius Header			

Appendix I   Guidelines for Version Number Assignment

Interoperability between different protocol versions is based on the following principles:

Robustness Principle:

RFC791 defines the general "robustness principle" for the internet protocol as follows:

- "An implementation must be conservative in its sending behavior and liberal in its receiving behavior".

Following such a robustness principle, it is possible to permit minor changes in the protocol while still maintaining backwards compatibility.

The general rule for protocol version numbering within PacketCable Multimedia Gate Control Protocol is as follows:

- Protocol versions within the same major version number MUST be backwards compatible. Versions with different Major Version Number are not expected to be backwards compatible.

It is crucial that the PacketCable Multimedia Specification team examine all protocol changes to be included in a new version of the protocol and select a protocol version number based on the change with the largest impact. If any of the changes satisfy the criteria for a major protocol version change, then the major version must be incremented.

Examples of protocol changes which would result in change to the minor version number:

- The introduction of a new optional object so long as the inclusion of the new object in a message does not introduce new mandatory functional requirements on the network element receiving the message such that the object could be safely ignored.

- The deprecation of an optional object

Examples of protocol changes which would result in a change to the major version number:

- The introduction of a new message

- A change in the format of a given object

- A grammar change which prohibited the inclusion of a given object in a given message

- A grammar change which made an object mandatory in a given message

- A grammar change which made an object optional in a message in which it was previously mandatory

- The introduction of a new optional object which when included in a message introduces new mandatory functional requirements on the network element receiving the message such that the new object could not be safely ignored.

- A semantic change to the protocol related algorithms or states (such as the gate state machine) that could result in a state inconsistency between devices running the new and old protocol versions.

Some changes, such as those which introduce new functionality, are difficult to classify. For example, one could imagine a change which introduced a new object and functional requirements on the network element receiving the object in a message. If the network element receiving the object was operating on a lower version of the protocol in which the new object was not defined, then the default behavior would be that the object would be ignored and consequently the behavior implied by the new object would not be performed. If the new behavior that is not performed due to the object being ignored was local to the receiving network element, one could argue that in this case, the two network elements are successfully interoperating at the lower version of the protocol. If, on the other hand, the presence of the new object in a message required the receiving network element to send a new response or to modify an existing response based on the new object, then to ignore the new object might prevent interoperability. In the latter case, a major version change would be required.

Other changes, such as those which change the status of an object within a message from mandatory to optional or vice versa could result in interoperable implementations based on the behavior of the sender. However, since the

behavior of a sender with respect to optional parameters cannot be guaranteed, such changes should be classified a major changes.

Given that many types of protocol changes would require a major protocol version change, it makes sense to group changes to the protocol such that major version changes occur infrequently and new versions provide significant value which justifies their implementation.

Appendix II   Acknowledgements

The CableLabs community wishes to heartily thank the following participants who have contributed directly to this document.

Contributor	Company Affiliation
Susie Riley	ADC
Dan Torbet	Arris
Doc Evans	Arris
Tim Keenan	Arris
Bill Hanks	Arris
Steve Shilling	Arris
Harindranath, P.R. Nair	C-Cor
Kevin Johns	CableLabs
Maria Stachelek	CableLabs
Eric Rosenfeld	CableLabs
Jeff Mandin	CableMatrix
Dan Haim	CableMatrix
John Pickens	CableMatrix
Matt Tooley	CableMatrix
Satish Putta	CableMatrix
Adam Smith	Camiant
Andy Stone	Camiant
Andy Bennett	Camiant
Geoff Devine	CedarPoint
Flemming Andreasen	Cisco
Brian Davis	Cisco
Sangeeta Ramakrishnan	Cisco
Jason Gaedtke	Comcast
Eric Smith	Ellacoya
Burcak Beser	Juniper
Dave Flanagan	Motorola
Shannon Urrutia	Nokia Siemens Networks
Tony MacDonald	Nortel
Andrew Sundelin	Stargus

Karthik Sundaresan, CableLabs

Appendix III Revision History

The following ECNs have been incorporated into PKT-SP-MM-I02-040930 and collectively represent Multimedia Protocol version 1.0.

ECN Identifier	Accepted Date	Title of EC	Author
MM-N-03.0115-3	9/13/2004	Renumber the Event Message IDs and Attribute IDs.	D. Flanagan
MM-N-03.0124-3	9/13/2004	Clarify requirements concerning the order of COPS and PCMM objects.	A. Stone
MM-N-03.0125-3	9/13/2004	The proper mapping of Traffic Profile envelope field to envelope specification set is unclear.	B. Hanks
MM-N-04.0139-2	9/13/2004	New Committed Recover State Upon T3 Expiry	B. Hanks
MM-N-04.0143-4	9/13/2004	Policy Server Event Messaging BCID and RKS clarification	G. Devine
MM-N-04.0146-3	9/13/2004	Add IANA assignments to the specification.	B. Davis
MM-N-04.0149-5	9/13/2004	Classifier Wild-Carding	B. Barker
MM-N-04.0151-3	9/13/2004	Clarify use of GateSpec:SessionClassID field, and allow policy servers to modify requests.	A. Stone
MM-N-04.0159-1	9/13/2004	Protocol ID Field in Classifier Object	B. Hanks
MM-N-04.0174-4	9/13/2004	Proper Handling of the T2 Timer	B. Barker
MM-N-04.0158-6	9/20/04	Editorial Cleanup	B. Davis
MM-N-04.0144-3	9/27/04	Technical changes to RTPS to DOCSIS mapping section	B. Davis
MM-N-04.0145-4	9/27/04	Adds versioning capabilities to the Multimedia specification	B. Davis
MM-N-04.0206-1	9/27/04	Policy Servers Modification Requests	A. Stone

The following ECNs have been incorporated in PKT-SP-MM-I03-051221.

ECN Identifier	Accepted Date	Title of EC	Author
MM-N-05.0223-3	11/7/05	UGS vs. RTPS usage clarification	Davis
MM-N-05.0224-3	11/7/05	Reserved ≥ Committed State corrections.	Davis
MM-N-05.0233-4	11/14/05	Downstream DSCP/TOS overwrite	Flanagan
MM-N-05.0234-5	11/21/05	Editorial	Johns
MM-N-05.0235-2	11/7/05	Event Messaging Alignment	Johns
MM-N-05.0236-8	11/21/05	Gate Control Message Error Code Generation	Pickens
MM-N-05.0237-2	11/7/05	Policy Denied Reason clarification	Bennett
MM-N-05.0238-3	11/14/05	Add Res ≥ Auth transition text in 6.2.2 to match state diagram	Flowers
MM-N-05.0239-6	11/21/05	Serial Handling of Objects	Pickens
MM-N-05.0240-3	11/14/05	Slack Term	Flanagan
MM-N-05.0241-2	11/7/05	T2 expiration in committed state	Flanagan
MM-N-05.0242-4	11/7/05	T3 Timer	Flanagan
MM-N-05.0243-4	11/21/05	Gate Time and Usage changes	Davis
MM-N-05.0270-2	11/7/05	Addition of Application Identifier to GateSpec	Smith
MM-N-05.0271-4	11/14/05	Service Class Name vs. Gate Spec parameters	Davis
MM-N-05.0283-4	11/21/05	MGPI Support	Johns
MM-N-05.0284-4	11/21/05	Align MM Classifier with DOCSIS Classifier	Smith
MM-N-05.0285-7	11/21/05	State Synch	Bennett

The following ECNs have been incorporated in PKT-SP-MM-I04-080522.

ECN Identifier	Accepted Date	Title of EC	Author
MM-N-06.0310-1	3/13/06	Update for Web Services Interface	Tooley
MM-N-07.0420-2	3/10/08	Support for user identifier	Bennett
MM-N-07.0421-2	3/10/08	Addition of Max Concatenated Burst to the BE, nrtPS and rtPS traffic profiles.	Davis
MM-N-07.0422-4	3/10/08	Handling of DOCSIS 3.0 Peak Rate TLV	Davis
MM-N-07.0423-3	3/10/08	PacketCable Multimedia References to DOCSIS Specification	MacDonald
MM-N-07.0428-2	3/10/08	Fixes to Detailed call flow based on D3.0 additions	Davis
MM-N-07.0429-2	3/10/08	DOCSIS 3.0 additions—Sequence Number, Segment Number, Attribute Mask	Flanagan
MM-N-07.0430-3	3/10/08	PacketCable Multimedia Support for IPv6	MacDonald
MM-N-07.0431-2	3/10/08	Update of Major/Minor version for I04	Johns
MM-N-07.0432-3	3/10/08	Traffic Profile: Upstream Drop	Urrutia
MM-N-07.0433-2	3/10/08	Removal of Open issues Section	Johns
MM-N-07.0467-1	3/10/08	CMTS Usage of Application Type	Johns
MM-N-08.0513-1	4/7/08	CMTS usage of SubscriberID	Davis

The following ECNs have been incorporated in PKT-SP-MM-I05-091029.

ECN Identifier	Accepted Date	Title of EC	Author
MM-N-09.0582-6	9/8/09	Multicast Support	Smith
MM-N-09.0590-1	10/19/09	Minor corrections to ECN 09.0582-6	Sundaresan

The following ECNs have been incorporated in PKT-SP-MM-I06-110715.

ECN Identifier	Accepted Date	Title of EC	Author
MM-N-10.0643-3	2/7/11	DSCP Overwrite Clarification	Sundaresan
MM-N-11.0660-2	3/14/11	PCMM Alignment with new DOCSIS SF parameters on Maximum Queue Depth	Sundaresan

The following ECN has been incorporated in PKT-SP-MM-I07-151111.

ECN Identifier	Accepted Date	Title of EC	Author
MM-N-13.0697-1	4/29/13	IPv4 / IPv6 Classifiers and Gates	Riddle

NOTE: This specification was closed in November 2019 and reopened for updates as version I08 via the engineering change process in November 2022. A PDF file of PKT-SP-MM-C01-191120 can be provided upon request.

The following ECNs have been incorporated in PKT-SP-MM-I09-230913.

ECN Identifier	Accepted Date	Title of EC	Author
MM-N-23.0704-2	3/16/2023	Modifying Provisioned SFs via PCMM	Ranganathan
MM-N-23.0705-3	4/6/2023	Persistent Gate Functionality	Medlock
MM-N-23.0706-3	4/20/2023	Traffic Profiles and PCMM Version	Sundaresan

✳ ✳ ✳