Data-Over-Cable Service Interface Specifications
Flexible MAC Architecture

FMA PacketCable™ Aggregator Interface Specification

CM-SP-FMA-PAI-I01-200930

ISSUED

DISCLAIMER

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

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

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

Document Status Sheet

Document Control Number:	CM-SP-FMA-PAI-I01-200930			
Document Title:	FMA PacketCable™ Aggregator Interface Specification			
Revision History:	W01 – 10/23/2019 D01 – 01/30/2020 D02 – 05/04/2020 D03 – 08/18/2020 I01 – 09/30/2020			
Date:	September 30, 2020			
Status:	~~Work in Progress~~	~~Draft~~	**Issued**	~~Closed~~
Distribution Restrictions:	~~Focus Team Only~~	~~CL/Member~~	~~CL/ Member/ Vendor~~	**Public**

Key to Document Status Codes

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

Trademarks

Contents

1 SCOPE 5
1.1 Introduction and Purpose 5
1.2 FMA Specification Documents 6
1.3 Requirements 6
2 REFERENCES 7
2.1 Normative References 7
2.2 Informative References 7
2.3 Reference Acquisition 7
3 TERMS AND DEFINITIONS 8
4 ABBREVIATIONS AND ACRONYMS 9
5 OVERVIEW 10
6 PACKETCABLE AGGREGATOR INTERFACE (PAI) 12
6.1 PAI Connection Management 12
6.2 PAI Example Message Flow 12
6.3 PAI Connectivity 13
6.3.1 TLS, Security 13
6.3.2 MAC NE-PAG Connection Establishment during Onboarding 14
6.3.3 Connection Management 14
6.3.4 Failure Processing 15
6.4 Protocol Information Elements and Functional Requirements 16
6.4.1 PAI Connection Management Messages 17
6.4.2 PAG-initiated DQoS and PCMM Signaling and Gate Control Messages 18
6.4.3 MAC NE-initiated DQoS and PCMM Signaling and Gate Control Messages 18
6.4.4 PCEM and PCMM Event Messages 19
6.4.5 Common PAI Protocol Requirements 20
6.5 Protocol Data Model 20
6.5.1 Implementation Guidelines 20
APPENDIX I ACKNOWLEDGEMENTS (INFORMATIVE) 22

Figures

Figure 1 - FMA with a PacketCable Aggregator 5
Figure 2 - PacketCable Aggregator and PAI Architecture 10
Figure 3 - Connection Management on PAI 12
Figure 4 - Example PCMM Session Setup using PAI 13
Figure 5 - Example Resynchronization after Connection Failure 16

Tables

Table 1 - List of FMA Specifications 6
Table 2 - PAI Connection Management Messages 17
Table 3 - PAG-initiated PAI DQoS and PCMM Signaling and Gate Control Messages 18
Table 4 - MAC NE-initiated PAI DQoS and PCMM Signaling and Control Messages 19
Table 5 - PAI Event Messages 19

1 SCOPE

1.1 Introduction and Purpose

Support for PacketCable is provided in FMA. A given FMA deployment could require PacketCable Dynamic Quality of Service (DQoS), PacketCable Multimedia (PCMM), PacketCable Event Messages (PCEM), any combination of the three, or no PacketCable at all. FMA provides all PacketCable functionality without impacting the legacy Call Management Server (CMS), PCMM Policy Server (PS), and PacketCable Record Keeping Server (RKS) applications nor the interfaces to them.

The FMA approach to PacketCable uses a PacketCable Aggregator (PAG) function to provide scale, operational simplicity, and security. The PAG, along with the FMA MAC Manager, gives the appearance to existing operator back-office systems and applications of a single I-CCAP, hiding the presence of subtended MAC Network Elements (MAC NEs). In this way, the CMS, Policy Server, and RKS connect to a PAG just as they would an I-CCAP. The PAG communicates to all subtended MAC NEs by way of a PacketCable Aggregator Interface (PAI). The PAI is the subject of this specification.

Figure 1 illustrates the PAG supporting one-to-many Remote MACPHY Device (RMD) MAC NEs. The PAG has a single PAI to each subtended MAC NE, providing the appropriate interface security and simplifying PacketCable connectivity significantly. Note that PAI is a textual shorthand for Pag-MacNe, the diagrammatic label for the PAI.

The PAG can be deployed co-located with the MAC Manager and can alternatively be deployed on a separate host. It is expected that in the first case the PAG and MAC Manager will have the same scope and manage the same set of MAC NEs. In the second case the PAG and MAC Manager can possibly manage different sets of MAC NEs. These alternative scenarios for deployment are currently implementation-specific in FMA and the interface between PAG and MAC Manager is unspecified.

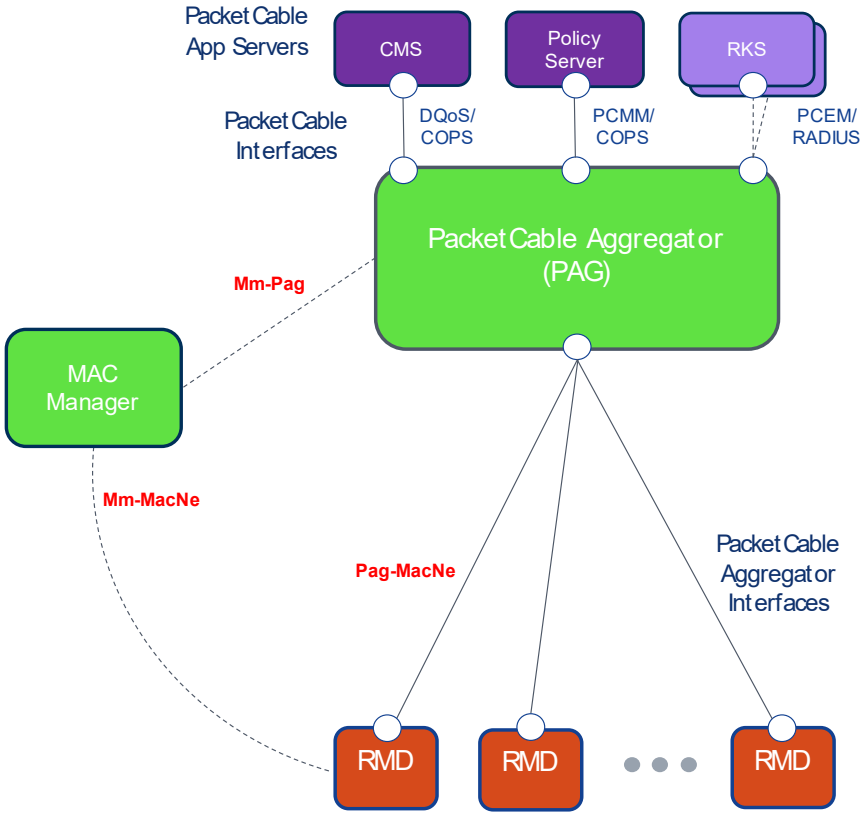

Figure 1 - FMA with a PacketCable Aggregator

1.2 FMA Specification Documents

A list of the documents in the FMA family of specifications is provided in Table 1. For the current issued versions of these specifications, refer to https://www.cablelabs.com/specifications.

Table 1 - List of FMA Specifications

Designation	Title
CM-SP-FMA-SYS	Flexible MAC Architecture System Specification
CM-SP-FMA-OSSI	Flexible MAC Architecture Operations Support System Interface Specification
CM-SP-FMA-MMI (this spec)	Flexible MAC Architecture MAC Manager Interface Specification
CM-SP-FMA-PAI	Flexible MAC Architecture PacketCable Aggregator Interface Specification

1.3 Requirements

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

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

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.

All references are subject to revision, and parties to agreement based on this specification are encouraged to investigate the possibility of applying the most recent editions of the documents listed below.

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

[FMA-MMI] Flexible MAC Architecture MAC Manager Interface Specification, CM-SP-FMA-MMI-I01-200930, September 30, 2020, Cable Television Laboratories, Inc.

[FMA-OSSI] Flexible MAC Architecture Operations Support Systems Interface Specification, CM-SP-FMA-OSSI-D01-200529, May 29, 2020, Cable Television Laboratories, Inc.

[FMA-SYS] Flexible MAC Architecture System Specification, CM-SP-FMA-SYS-I01-200930, September 30, 2020, Cable Television Laboratories, Inc.

[PAI-GPB] Google Protocol Buffers, http://mibs.cablelabs.com/GPB/DOCSIS/FMA/

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

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

2.2 Informative References

This specification uses the following informative references.

[PKT-ARCH2.0] PacketCable 2.0 Architecture Framework Technical Report, PKT-TR-ARCH-FRM-C01-140314, March 14, 2014, Cable Television Laboratories, Inc.

2.3 Reference Acquisition

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

3 TERMS AND DEFINITIONS

This specification uses the following terms.

cable modem	modulator-demodulator at subscriber locations intended for use in conveying data communications on a cable television system
downstream	(1) transmissions from CMTS to CM (2) RF spectrum used to transmit signals from a cable operator's headend or hub site to subscriber locations
GPB	Google Protocol Buffers
Internet Protocol (IP)	Internet network-layer protocol
MAC Manager	(1) management entity that represents a single point of contact for operator back-office management systems to FMA access network elements within its scope of control (2) FMA functional entity that uses FMA protocols to manage a MAC NE (e.g. RMD)
MAC Network Element (MAC NE)	FMA functional entity that contains DOCSIS MAC and upper layers functionality
Media Access Control (MAC)	refers to the Layer 2 element of the system, which would include DOCSIS framing and signaling
PacketCable Aggregator (PAG)	control plane entity that represents a single point of contact for operator back-office PacketCable and Policy Server systems to FMA access network elements within its scope of control
PacketCable Aggregator Interface (PAI)	control plane interface between a PAG and subtended MAC NEs.
Remote-MACPHY Device (RMD)	device in the network that implements the FMA specifications to provide conversion from digital Ethernet transport to analog RF transport
upstream	(1) transmissions from CM to CMTS (2) RF spectrum used to transmit signals from a subscriber location to a cable operator's headend or hub site

4 ABBREVIATIONS AND ACRONYMS

This specification uses the following abbreviations.

AM	Application Manager
CCAP	Converged Cable Access Platform
CM	cable modem
CMS	Call Management Server
CMTS	Cable Modem Termination System
COPS	Common Open Policy Service
DCA	Distributed CCAP Architecture
DF	Delivery Function
DOCSIS	Data-Over-Cable Service Interface Specifications
DQoS	Dynamic Quality of Service
FMA	Flexible MAC Architecture
FSM	finite state machine
I-CCAP	integrated CCAP
ID	Identifier
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
MAC	Media Access Control
MAC NE	MAC Network Element
MACPHY	Media Access Control and physical layers
MMI	MAC Manager Interface
NE	network element
OSS	operations support system
OSSI	Operations System Support Interface
PAG	PacketCable Aggregator
PAI	PacketCable Aggregator Interface
PCB	PAI Protocol Buffers
PCEM	PacketCable Event Messages
PCMM	PacketCable Multimedia
PS	Policy Server
QoS	quality of service
R-MACPHY	Remote MACPHY
RADIUS	Remote Authentication Dial-In User Service
RF	radio frequency
RKS	Record Keeping Server
RMD	Remote MACPHY Device
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol

5 OVERVIEW

[FMA-SYS] specifies requirements for PacketCable support in FMA. This section overviews the FMA PacketCable architecture to provide a context for specification of PacketCable Aggregator Interface (PAI) requirements in this document.

Figure 2 illustrates the FMA PacketCable architecture. The PacketCable Aggregator (PAG) provides northbound client-side interfaces for PacketCable Dynamic Quality of Service (DQoS) to the Call Management Server (CMS), for PacketCable Multimedia (PCMM) to the Policy Server, and for PacketCable Event Messages (PCEM) to the primary and secondary Record Keeping Server (RKS). In so doing, the PAG implements the COPS TCP client interfaces for DQoS and PCMM using the standard TCP ports for each protocol. The UDP client port for the RKS RADIUS connections is selected by the PAG.

The PAG contains functionality such as Message Buffering of southbound and northbound messages, Connection Management for COPS, RADIUS, and PAI connections, a Local Subscriber Database for mapping of subscribers to MAC NEs, and a Gate Database to map COPS Gates to and from PAI (i.e., per MAC NE) Gates. These functions are implementation-specific.

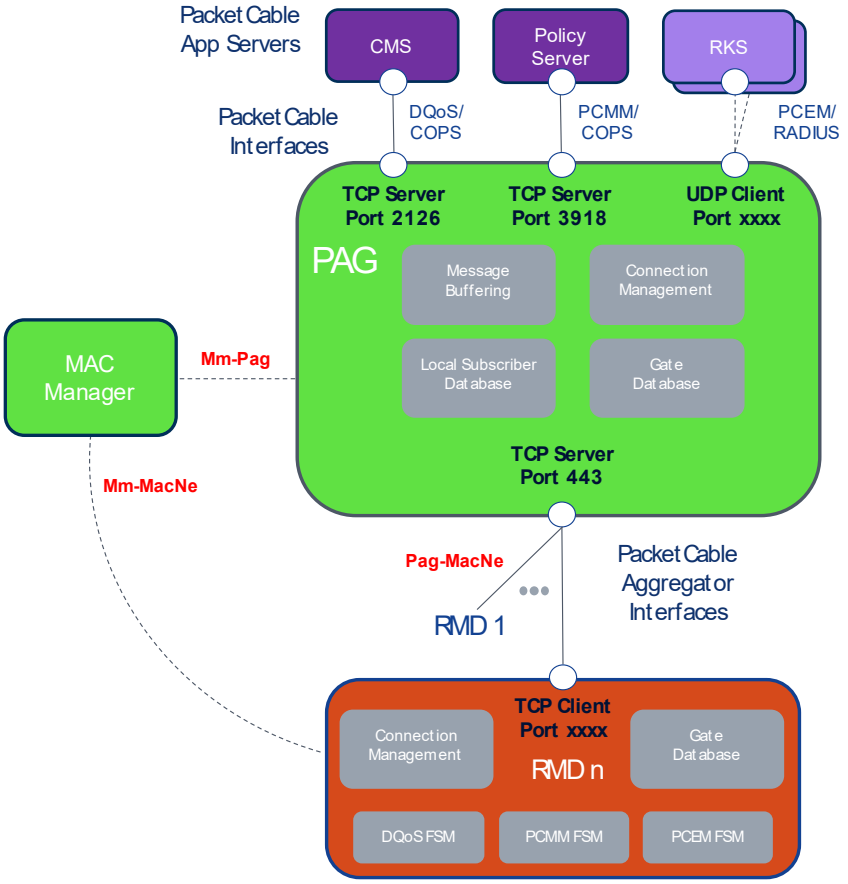

Figure 2 - PacketCable Aggregator and PAI Architecture

Southbound of the PAG are individual PAIs to each subtended MAC NE, which are in this example a number of Remote MACPHY Devices (RMDs). Note that PAI is a textual shorthand for Pag-MacNe, the diagrammatic label for the PAI. The PAG consolidates the transaction-based message sets from its northbound COPS and RADIUS interfaces onto a single transaction-based PAI per MAC NE. Otherwise, the PAG appears to be a CMS, Policy Server, and primary/secondary RKS from the perspective of the MAC NE.

Between each MAC NE and its configured PAG, a PAI is established to implement Gate-based operations that are requested from the PacketCable CMS and PCMM Policy Server. The PAG translates COPS based requests for each given subscriber device into equivalent PAI message. The PAG then determines which MAC NE the Gate operation needs to be initiated on and uses the PAI associated with that MAC NE to implement the operation. The MAC NE returns any status and event messaging resulting from the Gate operation to the PAG over its PAI.

The MAC NE contains certain types of functionality such as Connection Management for its PAI connection, FSMs for DQoS, PCMM and PCEM transactions, and a Gate Database for its internal Gate management. These functions are implementation-specific. However, the normative requirements for their operation are specified in [DQOS1.5], [PKT-MM], and [PKT-EM1.5]. In the FMA context, the MAC NE assumes the CMTS functionality required in those specifications.

There is an implementation-specific interface (Mm-Pag) between the MAC Manager and the PAG. This interface can be used by the MAC Manager to configure the PAG with [FMA-OSSI]-specified DOCSIS PacketCable configuration. It can also be used by the PAG to access the master Subscriber Database, which is populated by the MAC Manager. The Subscriber Database has, among other things, the subscriber device to MAC NE mappings necessary for the PAG to route southbound PAI messages.

6 PACKETCABLE AGGREGATOR INTERFACE (PAI)

6.1 PAI Connection Management

The MAC NE initiates PAI connection with its configured PAG during MAC NE onboarding and initialization, as described in [FMA-SYS]. The PAI is Transport Layer Security (TLS)-based. [FMA-SYS] specifies the various steps involved in TLS session establishment and MAC NE connection to the PAG.

Subsequent to TLS connection establishment and MAC NE connection to the PAG, PAI initialization and ongoing connection management on the PAI mimics the analogous operations that are done over a DQoS/PCMM COPS interface between the CMS/Policy Server and the PAG. The MAC NE sends a PAI ClientOpen message with the MAC NE ID and PAI version info parameters. The PAG responds with a PAI ClientAccept with the connection management KeepAliveMaxInterval. At this point the PAI is considered fully initialized and operational.

PAI KeepAlive messages are then periodically initiated by the MAC NE at an interval more frequent than specified in the KeepAliveMaxInterval. When a KeepAlive message is sent, the MAC NE starts a KeepAliveAck timer to time the wait for a response from the PAG. KeepAlive messages are responded to immediately by the PAG with PAI KeepAliveAck messages. On receipt of a PAG KeepAliveAck the MAC NE stops its KeepAliveAck timer.

On failure to receive a KeepAliveAck prior to expiry of the KeepAliveAck timer, the MAC NE logs an event.

Figure 3 illustrates PAI initialization and connection management for an RMD MAC Network Element. Normative connectivity requirements for the PAI are specified in Section 6.1.

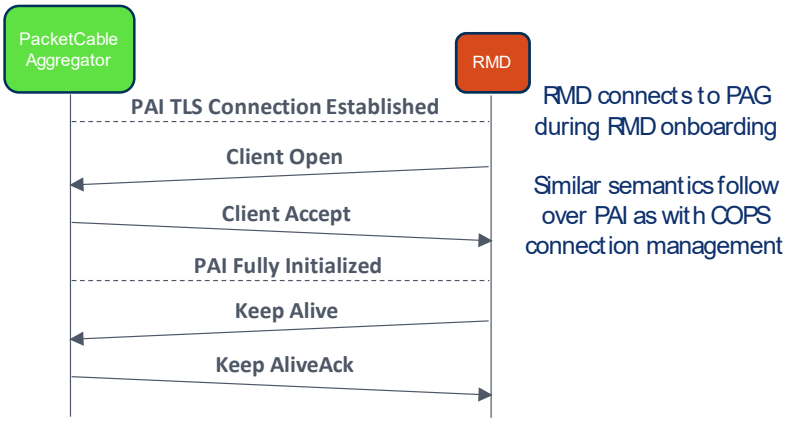

Figure 3 - Connection Management on PAI

6.2 PAI Example Message Flow

Figure 4 shows an example message flow illustrating a PCMM session setup initiated by client device signaling to a PacketCable Application Manager. The Application Manager sends a Gate-Set to the PacketCable Policy Server, which provides authorization for the session and forwards the Gate-Set to the PAG over the PCMM COPS interface. The PAG accesses its Local Subscriber Database to locate the RMD MAC NE where the client is homed, translates the Gate-Set into a PAI PcmmGateSet, and forwards it to the RMD over the PAI.

The RMD processes the PcmmGateSet by establishing a dynamic service flow with the client's Cable Modem (CM) using DOCSIS MAC signaling. The RMD assigns a GateID to the associated Gate and acknowledges PcmmGateSet with a PcmmGateSetAck containing the GateID. The RMD processing in this case is that of a CMTS, as specified in [PKT-MM], except that the RMD is using a PAI to the PAG instead of a COPS interface to the Policy Server.

The PAG processes the PcmmGateSetAck by recording the RMD-assigned GateID in its Gate Database. Since it is possible that a given GateID could be assigned by more than one connected RMD at any given point in time, the PAG implements an implementation-specific translation function to create a northbound GateID that is unique on a given COPS interface. The PAG translates the PcmmGateSetAck into a COPS Gate-Set-Ack, including the

northbound GateID, and transmits it to the Policy Server over the PCMM COPS interface. The Policy Server relays the Gate-Set-Ack to the Application Manager.

In this example, PacketCable Multimedia Event Messages are generated. The Application Manager has authorized a single stage reserve and commit of QoS resources, so the RMD follows its PcmmGateSetAck with both PcmmQosReserve and QosCommit event messages. RMD processing in this case is that of a CMTS, as specified in [PKT-MM], except that the RMD uses PAI to the PAG instead of RADIUS to the Primary RKS, and does not do failover processing to a Secondary RKS. The RMD waits for a PaiAck from the PAG after sending the PcmmQosReserve and before sending QosCommit. The RMD waits for a PaiAck from the PAG after sending the QosCommit and before sending any additional event messages.

The PAG processes the PAI PcmmQosReserve and QosCommit by translating them into RADIUS-based QoS-Reserve and QoS-Commit, respectively, and transmitting them to the Primary RKS. The PAG acknowledges receipt of PAI PcmmQosReserve and QosCommit with a PaiAck for each. The PAG keeps necessary message details until the messages are acknowledged by the RKS. The PAG tracks receipt of RKS message acknowledgements and performs failover processing between Primary and Secondary RKS per the rules of [PKT-MM], shielding the RMD from RKS failover responsibility.

The Primary RKS acknowledges receipt of the QoS-Reserve and QoS-Commit messages with QoS-Reserve-Ack and QoS-Commit-Ack responses (actually RADIUS Accounting-Responses) to the PAG. These acknowledgements prevent the PAG from resending the event messages and from performing RKS failover operations.

The end-to-end Application Signaling and Policy-Request/Policy-Request-Ack between the Policy Server and Application Manager are included for completeness but are out-of-scope of the FMA specifications.

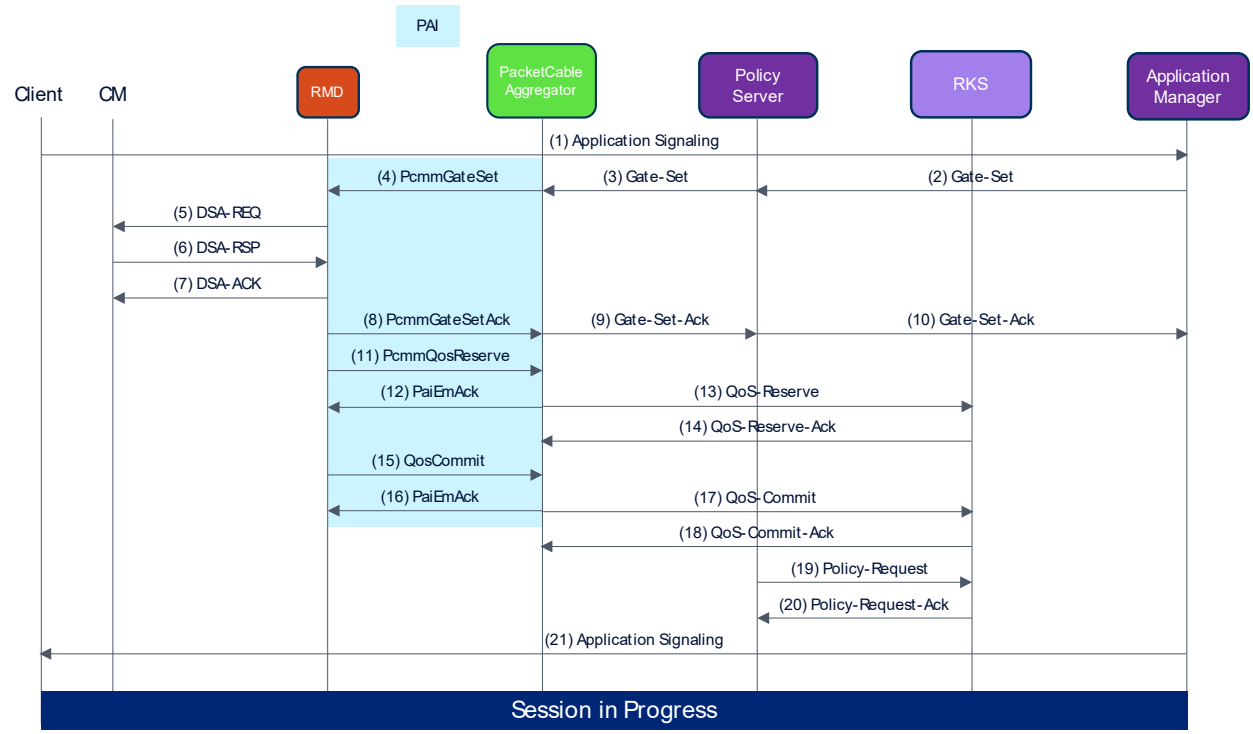

Figure 4 - Example PCMM Session Setup using PAI

6.3 PAI Connectivity

6.3.1 TLS, Security

The PAI is secured using TLS. Functional requirements related to the establishment of TLS sessions between the PAG and MAC NE during MAC NE onboarding are specified in [FMA-SYS].

6.3.2 MAC NE-PAG Connection Establishment during Onboarding

FMA defines three phases of MAC NE-PAG PAI connection establishment during MAC NE onboarding:

1. MAC NE obtaining its PAG IP address and TCP port, driven by the MAC NE and as specified in [FMA-SYS].
2. MAC NE-PAG TCP connection and TLS session establishment, driven by the MAC NE and specified in [FMA-SYS].
3. PAI initialization, driven by the MAC NE and specified below.

PAI initialization (reference Figure 3 above) consists of the MAC NE sending a PAI ClientOpen message to the PAG and receiving either a ClientAccept or ClientClose from the PAG in response. The MAC NE MUST initialize the PAI by sending a ClientOpen message to the PAG that includes its MAC NE ID and the supported PAI protocol version, as specified in [PAI-GPB]. When sending the ClientOpen message to the PAG, the MAC NE MUST start a PAI initialization timer. If the PAI initialization timer expires and the MAC NE has retries left per the configured PAI initialization retry count, the MAC NE MUST resend the ClientOpen message and restart the PAI initialization timer. If the PAI initialization timer expires and the MAC NE does not have retries left, the MAC NE MUST abort the initialization attempt, log a PAI initialization failure no ClientAccept critical event (70070027), and tear down the TLS connection to the PAG.

If the PAG receives a PAI ClientOpen message containing a MAC NE ID and a supported PAI version the PAG MUST respond to the MAC NE with a PAI ClientAccept message containing a KeepAliveMaxInterval and log a PAI initialization success notice event (70090101).

If the MAC NE receives a ClientAccept message from the PAG, the MAC NE MUST cancel the PAI initialization timer, start a timer for maximum interval between KeepAlive messages using ClientOpen KeepAliveMaxInterval, format and send a PAI KeepAlive to the PAG, and log a PAI initialization success notice event (70070030). When sending PAI KeepAlive to the PAG, the MAC NE MUST start a KeepAliveAck timer to verify that a PAG response to the MAC NE KeepAlive is received in a timely fashion.

If the PAG receives a PAI ClientOpen message with a PAI version that is not a version that the PAG can support, the PAG MUST respond with a PAI ClientClose message containing a CCR_UNSUPPORTED_PAI_VERSION ClientClose Reason. When sending a ClientClose containing CCR_UNSUPPORTED_PAI_VERSION, the PAG MUST include one if its own supported PAI versions in the message to help with subsequent PAI protocol version negotiations.

If it receives a ClientClose message from the PAG with a CCR_UNSUPPORTED_PAI_VERSION ClientCloseReason, and the MAC NE supports PAI versions that it has not yet tried in a ClientOpen, the MAC NE MUST send a new ClientOpen message with an untried PAI version and restart its PAI initialization timer. In this case, the MAC NE SHOULD consider the hint provided by the PAG in the ClientClose PAG PAI version field when selecting from its supported PAI versions. If it receives a ClientClose message from the PAG with a ClientCloseReason of CCR_UNSUPPORTED_PAI_VERSION, and the MAC NE has no untried PAI versions remaining, the MAC NE MUST send a ClientOpen message to the PAG with PAI version equal to zero to tell the PAG that PAI version negotiation has failed, and restart its PAI initialization timer to control possible retransmissions of the ClientOpen message, as specified above.

If it receives a PAI ClientOpen message with a MAC NE PAI version equal to zero, the PAG MUST respond with a PAI ClientClose message with ClientCloseReason of CCR_PAI_VERSION_NEGOTIATION_FAILURE, abort the PAI initialization, and log a PAI initialization failure PAI version negotiation critical event (70090104).

If the MAC NE receives a PAI ClientClose message with a CCR_PAI_VERSION_NEGOTIATION_FAILURE ClientClose Reason, the MAC NE MUST cancel the PAI initialization timer, abort the initialization, log a PAI initialization failure PAI version negotiation critical event (70070029), and tear down the TLS connection to the PAG.

6.3.3 Connection Management

The MAC NE MUST send PAI KeepAlive messages to the PAG at an interval the same as or more frequent than specified in the KeepAliveMaxInterval. The MAC NE MUST send each new PAI KeepAlive message with a monotonically increasing SequenceNumber. When transmitting a PAI KeepAlive message the MAC NE MUST

start a KeepAliveAck timer to verify that a PAG response to the MAC NE KeepAlive is received in a timely fashion.

The PAG MUST respond to a PAI KeepAlive message with PAI KeepAliveAck message.

On expiry of its KeepAliveAck timer without receiving a KeepAliveAck message, the MAC NE MUST log a Keep Alive ACK missed warning event (70070304).

6.3.4 Failure Processing

6.3.4.1 PAI Connection Failure Detection

The PAG detects PAI connection failure via failure to receive PAI KeepAlive messages from the MAC NE after successful PAI connection and initialization.

The MAC NE detects PAI connection failure via failure to receive a PAI KeepAliveAck message from the PAG after three consecutive expirations of the KeepAliveAck timer.

6.3.4.2 PAI Connection Failure Processing – PAG-detected

If the PAG determines that a MAC NE is no longer reachable (e.g. the PAG is no longer receiving PAI KeepAlive messages from the MAC NE), the PAG MUST log an PAI connection failure error event (70090205). If the PAG determines that a MAC NE is no longer reachable, the PAG MUST await PAI connection re-establishment, initiated by the MAC NE.

When the PAG loses connectivity with a given MAC NE, the PAG MUST reject any new Gate operations for subscriber devices associated with that MAC NE until the PAI connection is re-established. If the PAG loses connectivity with a given MAC NE, the PAG MUST reject any changes or status requests for established Gates for subscriber devices associated with that MAC NE until the PAI connection is re-established.

Once the PAI connection is reestablished following connection failure the PAG MUST log a PAI connection failure cleared notice event (70090206) and initiate post-failure PAG/MAC NE resynchronization procedures.

6.3.4.3 PAI Connection Failure Processing – MAC NE-detected

If it detects PAI connection failure, the MAC NE MUST keep all established Gates in place. If it detects PAI connection failure, the MAC NE MUST log a PAI connection failure error (70070305) event. If it detects PAI connection failure, the MAC NE MUST attempt to reconnect to the configured PAG up to the configured number of retries.

If it detects PAI connection failure either at the TCP or TLS session level or by failure to receive PAI KeepAliveAck messages from the PAG, the MAC NE MUST perform TLS session reestablishment by attempting a TCP connect to the PAG IP address and TCP port. If TCP connect to the PAG fails, the MAC NE MUST retry the connection up to the PAI connection retry count or until a successful TCP connection establishment, whichever comes first. If, after exhausting the number of PAI connection retries the MAC NE has not successfully established a TCP connection with the PAG, the MAC NE MUST abort the connection attempt and log a PAI connection failure error event (70070305).

After reestablishing a TCP connection with the PAG, the MAC NE MUST reestablish its PAG TLS session by performing TLS session establishment negotiations with the PAG. If TLS session establishment with the PAG fails, the MAC NE MUST retry the connection up to the PAI TLS session establishment retry count or until a successful TLS session establishment, whichever comes first. If, after exhausting the number of TLS session establishment retries the MAC NE has not successfully established a TLS session with the PAG, the MAC NE MUST abort the connection attempt and log a PAI connection failure error event (70070305).

When the PAI connection is restored, the MAC NE MUST log PAI connection failure cleared notice event (70070306) and notify the MAC Manager as specified in [FMA-MMI].

6.3.4.4 Connection Failure Processing Example – Resynchronization

An example of resynch after failure of an Application Manager to Policy Server COPS connection is shown in 5. The PAI plays a part in the end-to-end resynchronization after this type of failure, which is driven by the Application Manager upon restoration of the Application Manager to Policy Server COPS connection. The resynch is initiated with a PCMM Synch-Request from the Application Manager to the Policy Server. This Synch-Request propagates onward from the Policy Server to the PAG, where it is converted into a PAI PcmmSynchRequest and sent to the MAC NE, which in this example is an RMD.

The MAC NE responds with a sequence of one or more PcmmSynchReports, one per Gate, based on matching the request filters (i.e., AMID, PSID, SubscriberID) in the PcmmSynchRequest over the PAI. The PAG converts the PcmmSynchReports into PCMM Synch-Reports, which are propagated through the Policy Server to the Application Manager over the respective COPS interface. At the conclusion of the resynchronization the MAC NE sends a PcmmSynchComplete over the PAI. The PAG converts the PcmmSynchComplete into a PCMM Synch-Complete, which is propagated through the Policy Server to the Application Manager.

In the case where the failure is local to the PAI and the PAG determines that there are no northbound out-of-sync conditions, the resynchronization can be limited to the PAI messaging shown in Figure 5. When the failure is local to the PAI and the PAG determines that there is a possible northbound out-of-sync condition it can leverage PCMM protocol mechanisms to resynchronize network elements north of it. For example, the PAG can generate PCMM Gate-Report-State messages to inform both the Policy Server and Application Manager of changed state for a given Gate.

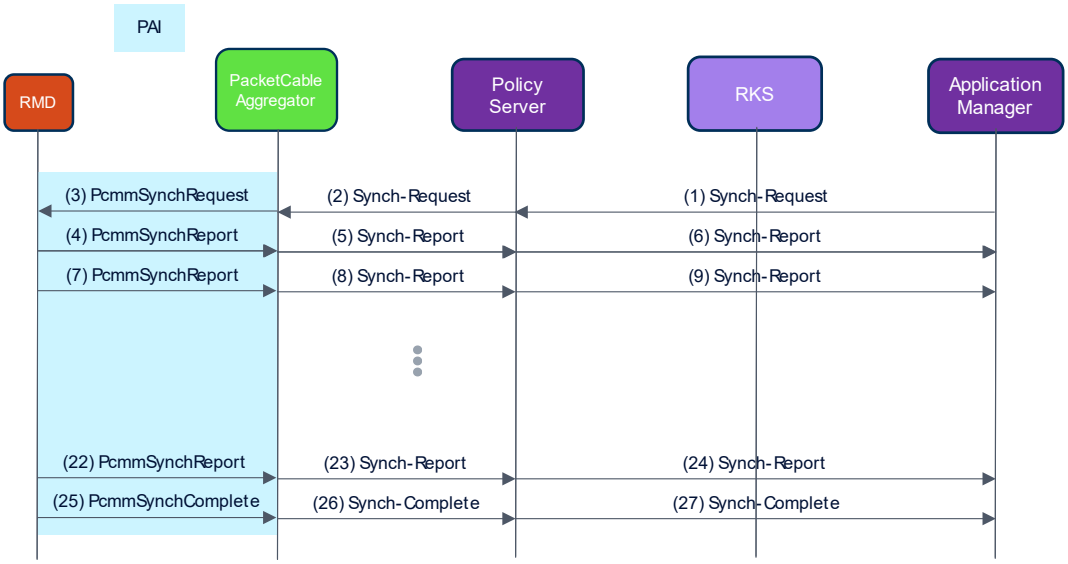

Figure 5 - Example Resynchronization after Connection Failure

6.4 Protocol Information Elements and Functional Requirements

The PAI protocol information elements are the message parameters and message definitions specified in [DQOS1.5], [PKT-MM], and [PKT-EM1.5]. PAI information elements as specified do not describe the message format used on the wire. That is left to the PAI protocol data model specified in [PAI-GPB].

Generally speaking, COPS and RADIUS overhead (e.g., COPS Common Message Format, Common COPS Message Header) is not necessary on the PAI and is not included in the PAI data model. Conformance to exact layouts of COPS and RADIUS message headers and payload is also not required on the PAI. PAI data model type definitions will support the necessary information element parameters, values, and ranges, but layout binary is determined by the protocol data model specified in [PAI-GPB]. PAI data model message definitions mirror those in [DQOS1.5], [PKT-MM], and [PKT-EM1.5] and provide traceability back to the original requirements.

Below are the message sets that are supported on the PAI and the associated functional requirements on the PAG and MAC NE.

6.4.1 PAI Connection Management Messages

Table 2 lists the connection management messages that are required to be supported over the PAI in terms of their DQoS and PCMM specified message definitions. Client-Open and Keep-Alive messages are initiated by the MAC NE and responded to by the PAG. The Client-Close message can be a response to Client-Open or can be initiated by the PAG.

Actual PAI message definitions specified in [PAI-GPB] differ slightly from what is shown in the table. For example, "Client-Open" is in PAI defined as "ClientOpen". "Keep-Alive" is in PAI defined as "KeepAlive". The difference in definition is derived from the PAI data model syntactical requirements.

Table 2 - PAI Connection Management Messages

MAC NE -> PAG	PAG -> MAC NE
Client-Open	Client-Accept
	Client-Close
Keep-Alive	Keep-Alive

6.4.1.1 MAC NE PAI Protocol Requirements

The MAC NE MUST format MAC NE-initiated PAI connection management messages as specified in [PAI-GPB].

The MAC NE MUST parse PAG responses to MAC NE-initiated PAI connection management messages as specified in [PAI-GPB].

The MAC NE MUST parse PAG-initiated PAI connection management messages as specified in [PAI-GPB].

The MAC NE MUST respond to receipt of a PAI PcmmGateSet containing a PCMM version number it does not support by rejecting the request with a PAI PcmmGateSetErr message containing a PktCblError error_code 127 (other, unspecified error) and error_subcode of 0, and by logging a PCMM version mismatch error event (70070031).

The MAC NE MUST respond to receipt of a PAI PcmmGateInfo containing a PCMM version number it does not support by rejecting the request with a PAI PcmmGateInfoErr message containing a PktCblError error_code 127 (other, unspecified error) and error_subcode of 0, and by logging a PCMM version mismatch error event (70070031).

The MAC NE MUST respond to receipt of a PAI PcmmGateDelete containing a PCMM version number it does not support by rejecting the request with a PAI PcmmGateDeleteErr message containing a PktCblError error_code 127 (other, unspecified error) and error_subcode of 0, and by logging a PCMM version mismatch error event (70070031).

The MAC NE MUST respond to receipt of a PAI PcmmSynchRequest containing a PCMM version number it does not support by rejecting the request with a PAI PcmmSynchComplete message containing a PktCblError error_code 127 (other, unspecified error) and error_subcode of 0, and by logging a PCMM version mismatch error event (70070031).

The MAC NE MUST respond to receipt of a PAI PcmmPdpConfig containing a PCMM version number it does not support by rejecting the request with a PAI PcmmPdpConfigErr message containing a PktCblError error_code 127 (other, unspecified error) and error_subcode of 0, and by logging a PCMM version mismatch error event (70070031).

6.4.1.2 PacketCable Aggregator PAI Protocol Requirements

The PAG MUST parse MAC NE-initiated PAI connection management messages as specified in [PAI-GPB].

The PAG MUST format responses to MAC NE-initiated PAI connection management messages as specified in [PAI-GPB].

The PAG MUST format PAG-initiated PAI connection management messages as specified in [PAI-GPB].

The PAG MUST include the negotiated PCMM version number of the COPS interface on which a PCMM Gate-Set message is received in the corresponding PAI PcmmGateSet message sent to the MAC NE.

The PAG MUST include the negotiated PCMM version number of the COPS interface on which a PCMM Gate-Info message is received in the corresponding PAI PcmmGateInfo message sent to the MAC NE.

The PAG MUST include the negotiated PCMM version number of the COPS interface on which a PCMM Gate-Delete message is received in the corresponding PAI PcmmGateDelete message sent to the MAC NE.

The PAG MUST include the negotiated PCMM version number of the COPS interface on which a PCMM Synch-Request message is received in the corresponding PAI PcmmSynchRequest message sent to the MAC NE.

The PAG MUST include the negotiated PCMM version number of the COPS interface on which a PCMM PDP-Config message is received in the corresponding PAI PcmmPdpConfig message sent to the MAC NE.

6.4.2 PAG-initiated DQoS and PCMM Signaling and Gate Control Messages

Table 3 lists DQoS and PCMM signaling and Gate control messages that are required to be supported over the PAI. This table shows the messages initiated by the PAG and associated responses received from the MAC NE in terms of their DQoS and PCMM specified message definitions.

Actual PAI message definitions specified in [PAI-GPB] differ slightly from what is shown in the table. For example, DQoS and PCMM have protocol-specific Gate-Set messages that are shown in the table below generically as "Gate-Set" but which are reflected as independent "DqosGateSet" and "PcmmGetSet" in the PAI data model.

Table 3 - PAG-initiated PAI DQoS and PCMM Signaling and Gate Control Messages

PAG -> MAC NE	MAC NE -> PAG	DQoS	PCMM
Gate-Alloc	Gate-Alloc-Ack, Gate-Alloc-Err	X	
Gate-Set	Gate-Set-Ack, Gate-Set-Err	X	X
Gate-Info	Gate-Info-Ack, Gate-Info-Err	X	X
Gate-Delete	Gate-Delete-Ack, Gate-Delete-Err	X	X
Synch-Request	Synch-Report, Synch-Complete		X
PDP-Config	PDP-Config-Ack, PDP-Config-Err		X

6.4.2.1 PacketCable Aggregator PAI Protocol Requirements

The PAG MUST format PAG-initiated DQoS and PCMM PAI signaling and Gate control messages as specified in [PAI-GPB].

The PAG MUST parse MAC NE responses to PAG-initiated DQoS and PCMM PAI signaling and Gate control messages as specified in [PAI-GPB].

6.4.2.2 MAC NE PAI Protocol Requirements

The MAC NE MUST parse PAG-initiated DQoS and PCMM PAI signaling and Gate control messages as specified in [PAI-GPB].

The MAC NE MUST format responses to PAG-initiated DQoS and PCMM PAI signaling and control messages as specified in [PAI-GPB].

6.4.3 MAC NE-initiated DQoS and PCMM Signaling and Gate Control Messages

Table 4 lists DQoS and PCMM signaling and control messages that are required to be supported over the PAI. This table shows messages initiated by the MAC NE and associated responses received from the PAG in terms of their DQoS and PCMM specified message definitions.

Actual PAI message definitions specified in [PAI-GPB] differ slightly from what is shown in the table. For example, DQoS has a Gate-Open message but PCMM does not. Consequently "Gate-Open" is reflected as "DqosGateOpen" in the PAI data model.

Table 4 - MAC NE-initiated PAI DQoS and PCMM Signaling and Control Messages

MAC NE->PAG	PAG -> MAC NE	DQoS	PCMM
Gate-Open		X	
Gate-Close		X	
Timeout-Notify		X	
Gate-Report-State			X
Gate-Cmd-Err			X
Msg-Receipt-Key	Msg-Receipt (an ACK requested by Msg-Receipt-Key parameter in PCMM messages)		X

6.4.3.1 MAC NE PAI Protocol Requirements

The MAC NE MUST format MAC NE-initiated DQoS and PCMM PAI signaling and Gate control messages as specified in [PAI-GPB].

The MAC NE MUST parse PAG responses to MAC NE-initiated DQoS and PCMM PAI signaling and Gate control messages as specified in [PAI-GPB].

6.4.3.2 PacketCable Aggregator PAI Protocol Requirements

The PAG MUST parse MAC NE-initiated DQoS and PCMM PAI signaling and Gate control messages as specified in [PAI-GPB].

The PAG MUST format responses to MAC NE-initiated DQoS and PCMM PAI signaling and control messages as specified in [PAI-GPB].

6.4.4 PCEM and PCMM Event Messages

Table 5 lists the PCEM (i.e., associated with DQoS) and PCMM event messages that are required to be supported over the PAI in terms of their PCEM and PCMM specified message definitions. All messages are initiated by the MAC NE and responded to by the PAG.

Actual PAI message definitions specified in [PAI-GPB] differ slightly from what is shown in the table. For example, "QoS-Commit" is reflected as "PcemQosCommit" and "PcmmQosCommit" in the PAI data model.

The PAG will provide a PAI-specific ACK to correspond to the RADIUS ACK used in the protocol between the RKS and PAG.

Table 5 - PAI Event Messages

MAC NE -> PAG	PAG -> MAC NE
QoS-Reserve	PaiEm ACK
QoS-Commit	PaiEm ACK
QoS-Release	PaiEm ACK
Time-Change	PaiEm ACK

6.4.4.1 MAC NE PAI Protocol Requirements

The MAC NE MUST format PCEM and PCMM event messages as specified in [PAI-GPB].

The MAC NE MUST parse acknowledgements to PCEM and PCMM event messages as specified in [PAI-GPB].

6.4.4.2 PacketCable Aggregator PAI Protocol Requirements

The PAG MUST parse PCEM and PCMM event messages as specified in [PAI-GPB].

The PAG MUST format acknowledgements to PCEM and PCMM event messages as specified in [PAI-GPB].

6.4.5 Common PAI Protocol Requirements

The PAI protocol data model specifies Boolean existence flags for all optional message fields. Existence flags are suffixed with "_is_valid" and default to false.

6.4.5.1 PacketCable Aggregator PAI Protocol Requirements

The PAG MUST place a fully formatted PaiHeader at the beginning of every transmitted PAI message as specified in [PAI-GPB]. The PAG MUST expect and parse a PaiHeader at the beginning of every received PAI message as specified in [PAI-GPB].

The PAG MUST set the existence flag to true for every optional message field that is included when encoding PAI messages.

The PAG MUST read the existence flag for every optional message field when decoding PAI messages.

The PAG MUST ignore optional message fields for which the existence flag is set to false in received PAI messages.

6.4.5.2 MAC NE PAI Protocol Requirements

The MAC NE MUST place a fully formatted PaiHeader at the beginning of every transmitted PAI message as specified in [PAI-GPB]. The MAC NE MUST expect and parse a PaiHeader at the beginning of every received PAI message as specified in [PAI-GPB].

The MAC NE MUST set the existence flag to true for every optional message field that is included when encoding PAI messages.

The MAC NE MUST read the existence flag for every optional message field when decoding PAI messages.

The MAC NE MUST ignore optional message fields for which the existence flag is set to false in received PAI messages.

6.5 Protocol Data Model

The PAI protocol data model is based on Google Protocol Buffers. The PAI protocol data model can be compiled to automatically generate PAI message encoding, parsing, and field access methods for use in interoperable PAG and MAC NE implementations. Normative PAI protocol data model parameter and message definitions are in [PAI-GPB].

6.5.1 Implementation Guidelines

Practical use of Google Protocol Buffers has led to creation of the following guidelines, which are followed in the PAI protocol data model specification:

1. Optional message fields in the PAI data model will always be accompanied with an existence flag and it is required to set the existence flag at the encoder when the associated optional field is present. This is because optional fields that are not included in an encoded message result in the parsed object having the missing field set to its default value. Existence flags defined for optional fields are accompanied by generated existence access methods which will allow the parsing code to determine the presence or absence of the associated optional field clearly and concisely in the parsed message.
2. Nesting of protocol data model message definitions is limited due to the length of strings generated in access methods. This resulted in the need to abbreviate names in some instances in the PAI data model. In other instances, it results in a certain degree of repetition in the data model definition versus optimal

inheritance of message attributes. During vendor development and interoperability testing it may be necessary to further reduce nesting in the protocol data model definitions.

3. For backward compatibility, the following rules need to be followed when updating a .proto data model:
   a. Do not change the tag numbers of existing fields.
   b. New fields can be added but need fresh tag numbers (i.e., tag numbers that were never used in this .proto file, even by deleted fields).
   c. Starting in Protocol Buffers Version 3, all fields are optional and can be deleted.
   d. Use the “reserved” function to identify deprecated/deleted tag numbers.
   e. Use the “reserved” function to identify deprecated/deleted field names.

Appendix I Acknowledgements (Informative)

We wish to thank the following participants contributing directly to this document:

FMA Leadership Participants

At the time this specification was submitted to CableLabs for approval, the Flexible MAC Architecture (FMA) Working Group (WG) and Task Forces (TF) under the WG was organized with the following leadership roles and affiliations:

Flexible MAC Architecture (FMA) Working Group (WG)

FMA WG Co-Chair	Michael "Mike" Emmendorfer, CommScope Inc.
FMA WG Co-Chair	Jon Schnoor, Cable Television Laboratories, Inc.
FMA WG Co-Chair	Jeff Finkelstein, Cox Communications

Task Force 1: MAC Manager Interface

Task Force 1 Co-Chair	Douglas Johnson, Vecima Networks Inc.
Task Force 1 Co-Chair	Stephen Kraiman, CommScope Inc.

Task Force 2: PacketCable and Lawful Intercept

Task Force 2 Co-Chair	Rex Coldren, Vecima Networks Inc.
Task Force 2 Co-Chair	Dan Torbet, CommScope Inc.

Task Force 3: FMA OSS Interface Specification

Task Force 3 Co-Chair	Steve Burroughs, Cable Television Laboratories, Inc.
Task Force 3 Co-Chair	Dwain Frieh, CommScope Inc.

Task Force 4: Data Forwarding

Task Force 4 Co-Chair	Mike Patrick, Harmonic Inc.
Task Force 4 Co-Chair	Mircea Orban, CommScope Inc.

Task Force 5: FMA-Proactive Network Maintenance

Task Force 5 Co-Chair	Jason Rupe, Cable Television Laboratories, Inc.
Task Force 5 Co-Chair	Santhana Chari, CommScope Inc.

Contributor	Company Affiliation
Kirk Erichsen, Steve Goeringer, Jon Schnoor	CableLabs
Phillip Anderson	Charter
Mike Emmendorfer	CommScope
Steve Foley	CommScope
Ray Smith	CommScope
Dan Torbet	CommScope
Jeff Finkelstein	Cox
Raj Gujjari	Nokia
Andy James	Nokia
Trung Trinh	Nokia
Utku Yilmaz	Nokia
Rex Coldren	Vecima
Doug Johnson	Vecima

* * *