

TR-451

vOMCI Specification

Issue: 01 Amendment 1
Issue Date: February 2025

Notice

The Broadband Forum is a non-profit corporation organized to create guidelines for broadband network system development and deployment. This Technical Report has been approved by members of the Forum. This Technical Report is subject to change. This Technical Report is owned and copyrighted by the Broadband Forum, and all rights are reserved. Portions of this Technical Report may be owned and/or copyrighted by Broadband Forum members.

Intellectual Property

Recipients of this Technical Report are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of this Technical Report, or use of any software code normatively referenced in this Technical Report, and to provide supporting documentation.

Terms of Use

1. License

Broadband Forum hereby grants you the right, without charge, on a perpetual, non-exclusive and worldwide basis, to utilize the Technical Report for the purpose of developing, making, having made, using, marketing, importing, offering to sell or license, and selling or licensing, and to otherwise distribute, products complying with the Technical Report, in all cases subject to the conditions set forth in this notice and any relevant patent and other intellectual property rights of third parties (which may include members of Broadband Forum). This license grant does not include the right to sublicense, modify or create derivative works based upon the Technical Report except to the extent this Technical Report includes text implementable in computer code, in which case your right under this License to create and modify derivative works is limited to modifying and creating derivative works of such code. For the avoidance of doubt, except as qualified by the preceding sentence, products implementing this Technical Report are not deemed to be derivative works of the Technical Report.

2. NO WARRANTIES

THIS TECHNICAL REPORT IS BEING OFFERED WITHOUT ANY WARRANTY WHATSOEVER, AND IN PARTICULAR, ANY WARRANTY OF NONINFRINGEMENT AND ANY IMPLIED WARRANTIES ARE EXPRESSLY DISCLAIMED. ANY USE OF THIS TECHNICAL REPORT SHALL BE MADE ENTIRELY AT THE USER'S OR IMPLEMENTER'S OWN RISK, AND NEITHER THE BROADBAND FORUM, NOR ANY OF ITS MEMBERS OR SUBMITTERS, SHALL HAVE ANY LIABILITY WHATSOEVER TO ANY USER, IMPLEMENTER, OR THIRD PARTY FOR ANY DAMAGES OF ANY NATURE WHATSOEVER, DIRECTLY OR INDIRECTLY, ARISING FROM THE USE OF THIS TECHNICAL REPORT, INCLUDING BUT NOT LIMITED TO, ANY CONSEQUENTIAL, SPECIAL, PUNITIVE, INCIDENTAL, AND INDIRECT DAMAGES.

3. THIRD PARTY RIGHTS

Without limiting the generality of Section 2 above, BROADBAND FORUM ASSUMES NO RESPONSIBILITY TO COMPILE, CONFIRM, UPDATE OR MAKE PUBLIC ANY THIRD PARTY ASSERTIONS OF PATENT OR OTHER INTELLECTUAL PROPERTY RIGHTS THAT MIGHT NOW OR IN THE FUTURE BE INFRINGED BY AN IMPLEMENTATION OF THE TECHNICAL REPORT IN ITS CURRENT, OR IN ANY FUTURE FORM. IF ANY SUCH RIGHTS ARE DESCRIBED ON THE TECHNICAL REPORT, BROADBAND FORUM TAKES NO POSITION AS TO THE VALIDITY OR INVALIDITY OF SUCH ASSERTIONS, OR THAT ALL SUCH ASSERTIONS THAT HAVE OR MAY BE MADE ARE SO LISTED.

All copies of this Technical Report (or any portion hereof) must include the notices, legends, and other provisions set forth on this page.

Issue History

Issue Number	Issue Date	Issue Editor	Changes
1	7 June 2022	Tim Carey (Nokia)	Original
01 Amendment 1	3 February 2025	André Brízido (Altice Labs)	Added YANG for extended TCP endpoint configuration; Added YANG for TLS endpoint configuration; Added Protobuf service for improved performance; Added support for Bulk messages, PLOAM dependencies and Secure Mutual Authentication

Comments or questions about this Broadband Forum Technical Report should be directed to info@broadband-forum.org.

Editor(s)

André Brízido, Altice Labs

Project Stream Leader

Ken Kerpez, DZS

Work Area Director(s)

Mengmeng Li, China Mobile

Bruno Cornaglia, Vodafone

Haomian Zheng, Huawei

Acknowledgements

The editors wish to acknowledge the following individuals for their contributions towards the Technical Report and/or the associated YANG data models.

Technical Report:

André Brízido, Altice Labs
 Ilias Gravalos, Nokia
 Jeff Hartley, CommScope
 Ludwig Pauwels, Nokia
 Robert Peschi, Nokia
 Tim Carey, Nokia

YANG & Protobuf Models

André Brízido, Altice Labs
 Ilias Gravalos, Nokia
 Jeff Hartley, CommScope
 Joey Boyd, Adtran
 Ludwig Pauwels, Nokia
 Nick Hancock, Adtran
 Robert Peschi, Nokia
 Tim Carey, Nokia

TABLE OF CONTENTS

EXECUTIVE SUMMARY	9
1 PURPOSE AND SCOPE	10
1.1 SCOPE	10
2 REFERENCES AND TERMINOLOGY	13
2.1 CONVENTIONS	13
2.2 REFERENCES	13
2.2.1 <i>Published References</i>	13
2.3 DEFINITIONS	15
2.4 ABBREVIATIONS	15
2.5 FIGURE AND TABLE NUMBERING CONVENTION	16
3 TECHNICAL REPORT IMPACT	17
3.1 ENERGY EFFICIENCY	17
3.2 SECURITY	17
3.3 PRIVACY	17
4 OVERVIEW	18
5 VOMCI SOLUTION OVERVIEW	19
5.1 ONBOARDING OF VNFS/CNFS	19
5.2 VOMCI FUNCTION DESCRIPTION	19
5.2.1 <i>Onboarding the vOMCI Function</i>	20
5.2.2 <i>Management of the vOMCI Function</i>	21
5.2.3 <i>OMCI Request and Response Processing</i>	23
5.2.4 <i>OMCI Notification Processing</i>	25
5.2.5 <i>Performance Monitoring</i>	27
5.2.6 <i>PLOAM Dependencies</i>	28
5.2.7 <i>OLT – ONU Secure Mutual Authentication</i>	29
5.3 ONU MANAGEMENT PROXY	32
5.3.1 <i>Management of the ONU Management Proxy</i>	33
5.3.2 <i>General Requirements</i>	33
5.3.3 <i>VNF/CNF Requirements</i>	35
5.3.4 <i>vOLTMF to ONU Management Proxy Management Interface ($M_{vOLTMF-vOMCI}$)</i>	35
5.4 VOMCI PROXY	36
5.4.1 <i>Management of the vOMCI Proxy</i>	37
5.4.2 <i>General Requirements</i>	38
5.4.3 <i>VNF/CNF Requirements</i>	38
5.4.4 <i>vOLTMF to vOMCI Proxy Management Interface</i>	39
5.4.5 <i>PLOAM Service Relay</i>	39
5.4.6 <i>SMA-OMCI Relay</i>	40
5.5 OLT DESCRIPTION	40
5.5.1 <i>vOMCI Function Connectivity</i>	41
5.5.2 <i>OMCI Message Forwarding</i>	44
5.5.3 <i>vOMCI ONU Notifications</i>	45
5.5.4 <i>Management of the OLT</i>	46
5.5.5 <i>Expose PLOAM Service</i>	46
5.5.6 <i>Secure Mutual Authentication</i>	48
5.6 VOLT MANAGEMENT FUNCTION DESCRIPTION	48
5.6.1 <i>Requirements</i>	49
5.6.2 <i>ONU Alarm Management</i>	50
5.6.3 <i>ONU OMCI Topology Management</i>	51

5.6.4	ONU Persistent Configuration and State Management.....	53
5.7	VOLT MANAGEMENT FUNCTION TO VOMCI FUNCTION OR VOMCI PROXY INTERFACE (M _{VOLTMF-VOMCI})..	54
5.7.1	Information Model and Messages.....	54
5.7.2	Transmission Encoding.....	65
5.7.3	Message Transfer Protocol.....	65
5.7.4	Requirements.....	70
5.8	VOMCI FUNCTION TO OLT INTERFACE (M _{VOMCI-OLT}).....	71
5.8.1	vOMCI Messages.....	71
5.8.2	Encapsulating vOMCI Messages in GPB for use with gRPC Transport.....	84
5.8.3	OMCI Message Retransmission Capability	86
5.8.4	Use of gRPC to Exchange GPB Encapsulated vOMCI Messages.....	88
5.9	SMA – OMCI INTERFACE (VOMCI – SMA).....	90
5.9.1	Transport Mechanism	90
5.9.2	Messages Definition.....	90
5.9.3	Encapsulate Messages in gRPC Transport.....	96
ANNEX A:	VOMCI YANG MODULES	97
A.1	OLT YANG MODULES.....	97
A.2	VOLTMF YANG MODULES.....	98
A.3	VOMCI FUNCTION YANG MODULES.....	99
A.4	ONU MANAGEMENT PROXY FUNCTION YANG MODULES	100
A.5	VOMCI PROXY FUNCTION YANG MODULES	101
A.6	DEPENDENCIES ON RELATED YANG MODULES AND STANDARDS.....	102
APPENDIX I.	VOMCI DEPLOYMENT IN CLOUDCO	104
I.1	INTEGRATING THE VOMCI SOLUTION WITH THE BAA LAYER.....	104
I.2	THE VOMCI SOLUTION DEPLOYED AS STANDALONE NFVI	105
I.3	CLOUDCO FRAMEWORK WITH VOMCI SOLUTION.....	106
I.4	VOLTMF IMPLEMENTED AS AN ABSTRACTION LAYER BETWEEN THE OLT AND VOMCI FUNCTION.....	107
APPENDIX II.	VOMCI DEPLOYMENT IN FANS	111
II.1	MANAGEMENT SYSTEM MODEL	112
II.2	VIRTUAL ACCESS NODE MODEL	113
APPENDIX III.	INTEROPERABILITY OF VOMCI FUNCTIONALITY WITH MULTI-VENDOR ONUS...115	
III.1	MULTI-VENDOR ONU IOP WITH OMCI	115
III.2	MULTI-VENDOR ONU IOP WITH SINGLE VOMCI FUNCTION (1:N SCENARIO)	115
III.3	MULTI-VENDOR ONU IOP WITH SPECIFIC VOMCI FUNCTION (N*1:1 SCENARIO)	116
APPENDIX IV.	ONU AUTHENTICATION IN VARIOUS SCENARIOS.....117	
APPENDIX V.	SUPPORTING INTERACTIONS FOR VOMCI DEPLOYMENTS	118
V.1	CONFIGURED ONU IS DETECTED ONLINE	118
V.2	AUTOMATED ONU DISCOVERY	120
V.3	AUTOMATED ONU DISCOVERY FAILURE	122
V.4	SYNCHRONIZE PM-RELATED INTERVALS AND TIMER.....	124
V.5	BACKGROUND ONU SOFTWARE DOWNLOAD SUPPORT	126
APPENDIX VI.	CONNECTIVITY MANAGEMENT AMONG VOLTMF(S) AND VOMCI FUNCTIONS....129	
APPENDIX VII.	VOMCI FUNCTION CONNECTIVITY USING A SESSION PROXY	130
APPENDIX VIII.	EXAMPLES OF M_{VOLTMF-VOMCI} MESSAGES	131
VIII.1	ONU DEVICE DATA MODEL.....	131
VIII.1.1	Example ReplaceConfig Message for ONU Target.....	131

<i>VIII.1.2 Example Action Message to Set and ONU Management Chain</i>	<i>132</i>
APPENDIX IX. EXAMPLES FOR USING DATASTORE TAG	134
IX.1 USE OF DST IN CONFIGURATION REQUESTS.....	134
IX.2 USE OF DST WHEN THE VOMCI FUNCTION HAS RESTARTED	134
APPENDIX X. VOMCI-AS-A-SERVICE DEPLOYMENT	136
X.1 USE OF THE ONU MANAGEMENT PROXY	137
APPENDIX XI. INTRODUCTORY EXAMPLES FOR DEPLOYING VOLTMF AND VOMCI.....	138

List of Figures

Figure 1.1-1/Figure 1 Scope of the vOMCI Solution	12
Figure 5.2-1/Figure 2 vOMCI Function and vOMCI Proxy Interfaces.....	20
Figure 5.2-2 Secure Mutual Authentication end-to-end message flow	31
Figure 5.5-1/Figure 3 Multiple Session – OLT to vOMCI Connectivity.....	41
Figure 5.5-2/Figure 4 Multiple Session – vOMCI to OLT Connectivity (single vOMCI function).....	42
Figure 5.5-3/Figure 5 Multiple Session – vOMCI to OLT Connectivity (vendor specific vOMCI function)	42
Figure 5.5-4/Figure 6 Multiple Session – vOMCI to OLT Connectivity (vOMCI Proxy).....	43
Figure 5.7-1/Figure 7 vOLTMF-vOMCI Message	55
Figure 5.7-2/Figure 8 vOLTMF-vOMCI HeaderMessage	56
Figure 5.7-3/Figure 9 vOLTMF-vOMCI Message Body.....	56
Figure 5.7-4/Figure 10 vOLTMF-vOMCI Request and Response Messages	57
Figure 5.7-5/Figure 11 vOLTMF-vOMCI Hello Request and Response Messages	59
Figure 5.7-6/Figure 12 vOLTMF-vOMCI GetData Request and Response Messages.....	60
Figure 5.7-7/Figure 13 vOLTMF-vOMCI ReplaceConfig Request and Response Messages	60
Figure 5.7-8/Figure 14 vOLTMF-vOMCI UpdateConfig Request and Response Messages	62
Figure 5.7-9/Figure 15 vOLTMF-vOMCI RPC and Action Request and Response Messages	63
Figure 5.7-10/Figure 16 vOLTMF-vOMCI Status Message.....	64
Figure 5.7-11/Figure 17 vOLTMF-vOMCI Notification Message	65
Figure 5.7-12/Figure 18 vOLTMF-vOMCI VomciMessageNbi Service	68
Figure 5.8-1/Figure 19 vOMCI Message Definition	72
Figure 5.8-2/Figure 20 ONU Header Message.....	73
Figure 5.8-3/Figure 21 OmciPacket Message	74
Figure 5.8-4/Figure 22 ONU Notification	74
Figure 5.8-5/Figure 23 ONU Command	75
Figure 5.8-6/Figure 24 ONU Response to ONU Command	75
Figure 5.8-7/Figure 25 ToD Packet Message Command	76
Figure 5.8-8/Figure 26 vOMCI Message Error Message	78
Figure 5.8-9/Figure 27 OLT Information Message	79
Figure 5.8-10 OMCI Packet Bulk Message	80
Figure 5.8-11 ONU Command and Response using OMCI Packet Bulk Messages.....	80
Figure 5.8-12 BIP Request Message	81
Figure 5.8-13 BIP Response Message.....	82
Figure 5.8-14 Allocate/Deallocate Alloc-ID Messages	82
Figure 5.8-15 G-PON AES Encryption Messages.....	83
Figure 5.8-16 Response Status Message	83
Figure 5.8-17/Figure 28 vOMCI SBI Service V2.....	84
Figure 5.8-18/Figure 29 Hello vOMCI Service V2	85
Figure 5.8-19/Figure 30 vOMCI Message Service V2.....	85
Figure 5.8-20 vOMCI PLOAM gRPC Service.....	86
Figure 5.8-21 PLOAM Function Utilization	86
Figure 5.9-1: SMA – OMCI Messages Definition.....	91
Figure 5.9-2: OLT Challenge Message	92
Figure 5.9-3: OLT Challenge OMCI interactions	92
Figure 5.9-4: ONU Challenge Outcome	92
Figure 5.9-5: ONU Response to OLT Challenge.....	93
Figure 5.9-6: OLT Authentication Result	93
Figure 5.9-7: Set OLT Authentication Result at the ONU.....	93
Figure 5.9-8: ONU Authentication Status Notification and Retrieval.....	94
Figure 5.9-9: MSK Name Retrieval.....	94
Figure 5.9-10: SMA-OMCI Error Message	95
Figure 5.9-11: SMA-OMCI gRPC Service	96
Figure A-1/Figure 31 vOMCI YANG Modules.....	97
Figure A-2/Figure 32 OLT vOMCI YANG Modules.....	98

Figure A-3/Figure 33 vOLTMF YANG Modules	99
Figure A-4/Figure 34 vOMCI Function YANG Modules.....	100
Figure A-5/Figure 35 ONU Management Proxy YANG Modules	101
Figure A-6/Figure 36 vOMCI Proxy YANG Modules	102
Figure II-1/Figure 44 TR-370 Management System Model	111
Figure II-2/Figure 45 Virtual Access Node Model.....	112
Figure II-3/Figure 46 vOMCI Solution for FANS Management System Model.....	113
Figure II-4/Figure 47 vOMCI Solution for FANS Virtual Access Node Model	114
Figure III-1/Figure 48 ONU IOP with Single vOMCI Function (1:N scenario)	115
Figure III-2/Figure 49 ONU IOP with Specific vOMCI Functions (N*1:1 scenario).....	116
Figure V-1/Figure 50 Configured ONU is Detected Online	118
Figure V-2/Figure 51 Automated ONU Discovery	121
Figure V-3/Figure 52 Automated ONU Discovery Failure	123
Figure V-4/Figure 53 Synchronize PM intervals for a Discovered ONU.....	125
Figure V-5/Figure 54 Background ONU Software Download	127
Figure VI-1/Figure 55 Connectivity Among vOMCI functions and vOLTMF(s)	129
Figure VII-1/Figure 56 Single Session – OLT to vOMCI Connectivity.....	130
Figure IX-1/Figure 57 Use of DST in ONU Configuration.....	134
Figure IX-2/Figure 58 Use of DST in vOMCI Function Restart	135
Figure X-1/Figure 59 vOMCI-as-a-Service	136

List of Tables

Table 5.4-1 Adaptation of Bulk Messages by vOMCI proxy	37
Table 5.7-1/Table 1 Kafka Topic Categories	66

Executive Summary

This Technical Report provides the architecture, requirements and interface specifications for a virtualized OMCI (vOMCI) solution that moves the OMCI functionality that is traditionally embedded within Optical Line Termination (OLT) network elements into the Operator's network. This Technical Report supports various deployment models of the vOMCI solution where functions (i.e., OMCI translation function, OLT & ONU management functions) of the architecture can be deployed as a virtualized network functions and are expected to be used within Access domain SDN management and control solutions (e.g., Broadband Forum's CloudCO) or as a stand-alone process that can be deployed with existing management system solutions.

The vOMCI solution allows Operators and Service providers more flexibility in how they create, activate and maintain services associated with ONUs. Likewise, the vOMCI solution enables easier interoperability testing and onboarding the ONUs within the Operator's ecosystem and network.

1 Purpose and Scope

This Technical Report specifies the architecture for the vOMCI solution with requirements associated with the functions that comprise the vOMCI solution and specifications for the interfaces associated with the functions identified in this Technical Report to attenuate interoperability issues in the xGPON deployments. This Technical Report complements TR-413 [37] and TR-385 [35] specifications.

1.1 Scope

The scope of this Technical Report is to provide an architecture for the vOMCI solution depicted in Figure 1.1-1 providing requirements and interface specifications for the function identified in this section as well as the requirements for the OLT as they relate to the responsibilities identified in this section of the Technical Report.

The vOLT Management Function (vOLTMF) shown in Figure 1.1-1, is responsible for:

- Receiving YANG notifications from the OLT regarding the ONU presence states.
- Informing the other entities in the vOMCI solution that an ONU can be communicated with for ONU management purposes.
- Receiving management and control requests from the northbound SDN M&C.
- Generating and sending requests for management of the OLT via the interface with OLT.
- Generating and sending requests to target ONUs via the interface with the vOMCI function or ONU Management Proxy.
- Receiving from the northbound SDN M&C the information necessary to connect OLTs, vOMCI functions, vOMCI Proxies and/or ONU Management Proxies.
- Receiving from the northbound SDN M&C the information necessary to associate the vOMCI function with ONUs.
- Receiving from the northbound SDN M&C the mapping information to associate the ONU characteristics (i.e., vendor, software info) with the versions supported by vOMCI function instances.
- Receiving notifications including events and alarms from the OLT and vOMCI function, or ONU Management Proxy and sending notifications to the northbound SDN M&C.

When deployed in the vOMCI solution, the ONU Management Proxy shown in Figure 1.1-1, is responsible for:

- Forwarding the management requests that are targeted for ONU or vOMCI function from the vOLTMF to the vOMCI function instances.
- Correlating responses to the requests previously sent to the vOMCI function instances and forwarding the responses to the vOLTMF.
- Receiving notifications from a vOMCI function instances and forwarding the notification to the vOLTMF.
- Load balancing the ONU associations for management traffic across available vOMCI function instances.

The vOMCI function shown in Figure 1.1-1, is responsible for:

- Receiving from the vOLTMF or ONU Management Proxy management commands issued towards the target ONU(s).
- Translating the received management commands into OMCI managed entities (ME) and formatting them into OMCI messages compliant with ITU-T G.988.
- Encapsulating and sending the above OMCI messages to the target ONU. The messages will pass through the vOMCI Proxy(s), if required by the vOMCI deployment scenario, and the OLT the target ONU is attached to.
- Translating the OMCI messages received from an ONU (e.g., containing ONU's operational data) through the vOMCI Proxy(s), if required by the vOMCI deployment scenario, and the OLT the target ONU is attached to.
- Sending the management data and information generated from the received OMCI messages to the vOLTMF or ONU Management Proxy.

When deployed in the vOMCI architecture, the vOMCI Proxy shown in Figure 1.1-1, is responsible for:

- Forwarding the vOMCI message between OLTs and vOMCI functions.
- Optionally perform functions related to the forwarding (e.g., OMCI message retransmission) of the vOMCI message.

The OLT shown in Figure 1.1-1, is responsible for:

- Decapsulating and processing received vOMCI messages and sending them to the target ONU using the relevant PON OMCI channel.
- Process the OMCI messages received from the ONU and encapsulating them into vOMCI messages and send them to the vOMCI function or vOMCI Proxy.
- Transmitting PLOAM notifications to the vOLT Management Function.

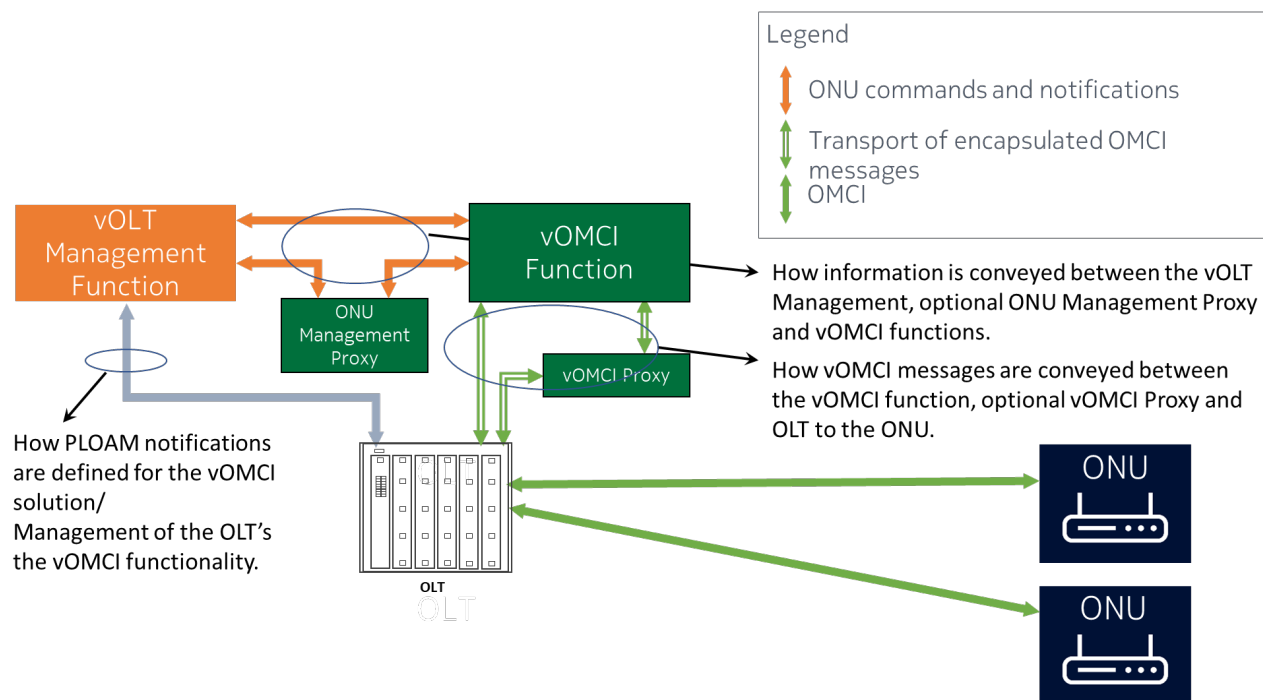


Figure 1.1-1/Figure 1 Scope of the vOMCI Solution

2 References and Terminology

2.1 Conventions

In this Technical Report, several words are used to signify the requirements of the specification and RFC 8174 [18]. These words are always capitalized. More information can be found in RFC 2119 [6]. The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [6] [RFC8174] [18] when, and only when, they appear in all capitals, as shown here.

2.2 References

2.2.1 Published References

The following references are of relevance to this Technical Report. At the time of publication, the editions indicated were valid. All references are subject to revision.

Users of this Technical Report are therefore strongly encouraged to investigate the possibility of applying the most recent edition of the listed references by checking at the web links reported below, especially for YANG DMs.

A list of currently valid BBF Technical Reports is published at www.broadband-forum.org/technical-reports. Users also may find useful BBF's searchable [Resources database](#) for Technical Reports and other BBF deliverables.

The most recent edition of BBF's published YANG Data Models can be found at [YANG Projects](#).

Document	Title	Source	Year
[1] G.988	<i>ONU management and control interface (OMCI) specification</i>	ITU-T	2017
[2] G.984.3	<i>Gigabit-capable passive optical networks (G-PON): Transmission convergence (TC) layer specification</i>	ITU-T	2014
[3] G.987.3	<i>10-Gigabit-capable passive optical networks (XG-PON): Transmission convergence (TC) layer specification</i>	ITU-T	2-14
[4] G.9807.1	<i>10-Gigabit-capable symmetric passive optical network (XGS-PON)</i>	ITU	2016
[5] G.989.3	<i>40-Gigabit-capable symmetric passive optical networks (NG-PON2): Transmission convergence layer specification</i>	ITU	2-15
[6] RFC 2119	<i>Key words for use in RFCs to Indicate Requirement Levels</i>	IETF	1997
[7] RFC 5246	<i>The Transport Layer Security (TLS) Protocol Version 1.2</i>	IETF	2008
[8] RFC 5905	<i>Network Time Protocol Version 4: Protocol and Algorithms Specification</i>	IETF	2010
[9] RFC 6125	<i>Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)</i>	IETF	2011

[10]	RFC 6241	<i>Network Configuration Protocol (NETCONF)</i>	IETF	2011
[11]	RFC 6991	<i>Common YANG Data Types</i>	IETF	2013
[12]	RFC 7317	<i>A YANG Data Model for System Management</i>	IETF	2014
[13]	RFC 7407	<i>A YANG Data Model for SNMP Configuration</i>	IETF	2014
[14]	RFC 7540	<i>Hypertext Transfer Protocol Version 2 (HTTP/2)</i>	IETF	2015
[15]	RFC 7895	<i>YANG Module Library</i>	IETF	2016
[16]	RFC 7950	<i>The YANG 1.1 Data Modeling Language</i>	IETF	2016
[17]	RFC 7951	<i>JSON Encoding of Data Modeled with YANG</i>	IETF	2016
[18]	RFC 8174	<i>Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words</i>	IETF	2017
[19]	RFC 8259	<i>The JavaScript Object Notation (JSON) Data Interchange Format</i>	IETF	2017
[20]	RFC 8528	<i>YANG Schema Mount</i>	IETF	2019
[21]	RFC 8632	<i>A YANG Data Model for Alarm Management</i>	IETF	2019
[22]	RFC 9640	<i>YANG Data Types and Groupings for Cryptography</i>	IETF	2024
[23]	RFC 9641	<i>A YANG Data Model for a Truststore</i>	IETF	2024
[24]	RFC 9642	<i>A YANG Data Model for a Keystore</i>	IETF	2024
[25]	RFC 9643	<i>YANG Groupings for TCP Clients and TCP Servers</i>	IETF	2024
[26]	RFC 9645	<i>YANG Groupings for TLS Clients and TLS Servers</i>	IETF	2024
[27]	ATP-247i3	<i>GPON & XG-PON1 ONU Conformance Abstract Test Plan</i>	BBF	2014
[28]	TR-101i2	<i>Migration to Ethernet-Based Broadband Aggregation</i>	BBF	2011
[29]	TR-156i4	<i>Using GPON Access in the context of TR-101</i>	BBF	2017
[30]	TR-167i3	<i>GPON-fed TR-101 Ethernet Access Node</i>	BBF	2017
[31]	TR-178i2	<i>Multi-service Broadband Network Architecture and Nodal Requirements</i>	BBF	2017
[32]	TR-280	<i>ITU-T PON in the context of TR-178</i>	BBF	2016
[33]	TR-383a7	<i>Common YANG Modules for Access Networks</i>	BBF	2023
[34]	TR-384	<i>Cloud Central Office Reference Architectural Framework</i>	BBF	2017
[35]	TR-385i2	<i>ITU-T PON YANG Modules, Issue 2</i>	BBF	2020
[36]	TR-370	<i>Fixed Access Network Sharing – Architecture and Nodal Requirements Issue 2</i>	BBF	2020
[37]	TR-413	<i>SDN Management and Control Interfaces for CloudCO Network Functions</i>	BBF	2019
[38]	Google Protobuf	Google Protobuf Language Guide (proto3)	GPB Open Source Community	NA
[39]	Google Remote Procedure Call	Google Remote Procedure Call (gRPC)	gRPC Open Source	NA

2.3 Definitions

The following terminology is used throughout this Technical Report.

vOMCI	Virtualization of the OMCI functionality within the PON Access Network
vOMCI Function	A function in the vOMCI solution that is responsible for the translation of ONU management requests, response and notification between OMCI messages and YANG objects.
vOMCI Service	The vOMCI-as-a-Service deployment organizes ONU Management Proxy, vOMCI Proxy and vOMCI function into a vOMCI service that is consumed by vOLTMF and other clients such as a slice management function.
vOLT Management Function	A function in the vOMCI solution that is responsible for the management of the PON Access Network that includes management of ONUs and OLTs. ONUs can be managed using the vOMCI solution or through the OMCI solution embedded within the OLT.
YANG	Data Modeling Language by IETF

2.4 Abbreviations

This Technical Report uses the following abbreviations:

AVC	Attribute Value Change
BAA layer	Broadband Access Abstraction layer
BBF	Broadband Forum
CloudCO	Cloud Central Office
CRC	Cyclic Redundancy Check
CRUD	Create, Retrieve, Update and Delete
DST	Datastore Tag
FANS	Fixed Access Network Sharing
GPB	Google Protocol Buffers
gRPC	Google RPC
HTTP	HyperText Transfer Protocol
InP	Infrastructure Network Provider
IOP	Interoperability
IPTV	Internet Protocol Television
JSON	JavaScript Object Notation
LOOC	Loss of OMCI Channel communication
MDS	MIB Data Synchronization
ME	Managed Entity
MIB	Management Information Base
MIC	Message Integrity Check
MSBN	Multi Service Broadband Network

M&C	Management and Control
NBI	Northbound Interface
NFVI	Network Function Virtualization Infrastructure
OAM	Operations Administration and Management
ODN	Optical Distribution Network
OMCC	ONU Management and Control Channel
OMCI	ONU Management and Control Interface
ONU	Optical Network Unit
OLT	Optical Line Termination
PLOAM	Physical Layer Operations, Administration and Maintenance
PON	Passive Optical Network
PMAA	Persistent Management Agent Aggregator
PNF	Physical Network Function
RFC	Request For Comments
RPC	Remote Procedure Call
vOLT	Virtualized OLT
vOLTMF	vOLT Management Function
SBI	Southbound Interface
SDN	Software Defined Network
SMA-OMCI	Secure Mutual Authentication – OMCI based
SN	Serial Number
SNMP	Simple Network Management Protocol
TCP	Transmission Control Protocol
ToD	Time of Day
TR	Technical Report
URL	Universal Resource Locator
VLAN	Virtual Local Area Network
VNF	Virtual Network Function
VNO	Virtual Network Operator
WA	Work Area
WT	Working Text

2.5 Figure and Table Numbering Convention

The Technical Report TR-451 used a sequential numbering for all figures and tables throughout the text. This convention changed starting from TR-451 Amendment 1. Now all figures are numbered as “Figure SN-FN”, whereas SN is the section number, FN is the figure number in the chapter. In cases where the figure was present in TR-451, the new figure number also includes ‘/Figure OLDFN’ to simplify tracking, whereas “OLDFN” is the figure number in TR-451. Similarly, all the tables are numbered as “Table SN-TN”, whereas TN is the table number in the chapter. If the table was present in TR-451, the new table number also includes ‘/Table OLDTN’ whereas “OLDTN” is the table number in TR-451.

3 Technical Report Impact

3.1 Energy Efficiency

This Technical Report may impact energy efficiency, as network functions can now be decoupled from existing standalone nodes. Use of generic hardware, as such not optimized for a specific network application, and migration of network functions to more distributed locations could lead to higher energy consumption. However, on demand allocation of hardware resources and hardware sharing across multiple applications can produce energy gains. This Technical Report does not intend to quantify these opposite effects on energy efficiency.

Regulatory differences related to electrical power, Heating, Ventilation and Air Conditioning (HVAC) and fire protection between traditional central offices and datacenters is out-of-scope for this document.

3.2 Security

Security provides "a form of protection where a separation is created between the assets and the threat." This Technical Report enables the sharing of a common infrastructure and interfaces between various use cases that may be operated by different departments (e.g., wireline and mobile) or different companies (other service providers, including other network service providers). This Technical Report also provides an increased opportunity for Operators to dynamically control the network service behavior, with the use of interfaces specified in this Technical Report.

It is noted that existing threats, safeguards, and enhancements remain applicable to this Technical Report's deployments in the management and control planes. This specification assumes a foundation of current security best practices that have been defined for the existing Multi Service Broadband Network (MSBN). However, some new or amplified concerns also appear and without appropriate precautions, the above conditions could impact a network's security.

3.3 Privacy

A multi-tenant vOMCI solution hosts functionality for a set of actors with potentially competing interests that the vOMCI solution will be required to isolate from each other. At the same time, it is required to enable business interactions between the same set of actors requiring careful design of the points of contact. A multi-tenant vOMCI solution is a system of sufficient complexity that will expose new attack vectors to malicious parties that have access to the system. For example, the "black box" steady state functionality of a virtualized system may be identical to a corresponding physical network function implementation, but the elasticity and dynamic behavior a virtualized system is capable of implies significantly different system responses to load will be possible, which can be exploited for malicious purposes if poorly designed or executed.

Privacy involves the need to ensure that information to, from and between customers can only be accessed by those who have the right to do so. Furthermore, privacy requirements can vary by regulatory region. In general, two ways to ensure privacy is recognized:

- Preventing data from being copied to a non-intended destination.
- Encrypting data, so that it cannot be understood even if it is intercepted.

This document does not define any specific mechanisms.

4 Overview

This section provides an overview of the vOMCI functionality in the context of various deployment scenarios (e.g., CloudCO, FANS) and the general purpose for the vOMCI interfaces.

Details of deployment scenarios are provided for information in some of the Appendixes.

5 vOMCI Solution Overview

The vOMCI solution assumes the Separated-NE mode described in TR-385 [35] where the OLT and the ONU are managed and configured independently. The vOMCI solution also assumes a northbound SDN M&C to manage the Access Network topology.

Note: The support of the two modes embedded OMCI (eOMCI) and vOMCI on the same management architecture is still under investigation.

The vOMCI solution disaggregates the ITU-T G.988 OMCI function traditionally hosted in the OLT into a virtualized network function called the vOMCI function hosted in the Service Provider's cloud platform. The vOMCI function is driven via well-defined CRUD (Create, Read, Update and Delete) primitives derived from YANG data models (i.e., TR-385 [35]) and performs the translation to the ITU-T G.988 OMCI [1].

By disaggregating the vOMCI function the following benefits can be achieved:

- PON deployment and operations improvements by decoupling the ONU management from the OLT. This is especially true when the Service Providers uses ONUs and OLTs from different vendors.
- Service agility is increased such that new capabilities can be introduced into the service provider's network at a faster rate than if the function remains within the OLT. The vOMCI solution is deployed in multiple deployment configurations including the Cloud Central Office (CloudCO) framework described in Appendix I and Fixed Access Network Sharing (FANS) described in Appendix II.

The vOMCI solution can optionally organize the ONU Management Proxy, vOMCI Proxy and the associated vOMCI function instances into a vOMCI service. The vOMCI service disaggregates from the vOLTMF the vOMCI service logic such as the association of ONUs and vOMCI function instances, ONU management chain setup, manipulation of the load balance and scale in/out of the vOMCI function instances. The complexity of the connectivity between entities (e.g., vOLTMF instance, OLT) that interaction with the vOMCI service is greatly reduced as the vOLTMF instances and OLTs no longer need to connect directly to each vOMCI function instance. Instead, the OLTs would connect to a vOMCI Proxy and the vOLTMF instance would connect to the ONU Management Proxy. Refer to Appendix X for the detailed description of the vOMCI service.

5.1 Onboarding of VNFs/CNFs

The deployment of vOLTMF, vOMCI functions, ONU Management Proxies and vOMCI Proxies as a VNF or a CNF requires additional considerations for onboarding in the Service Provider's NFV Infrastructure (NFVI) that eases the integration of the vOMCI solution into the Cloud environment (i.e., CloudCO BAA layer, CloudCO infrastructure). Minimally this means that the VNFs/CNFs have to be easily integrated in an Openstack or a Kubernetes infrastructure and implies that a VNF/CNF requires the associated descriptors and artifacts to ease its integration in the NFVI using well known tools such as Ansible, Heat and Helm. As typical of NFV deployments multiple instances of those VNFs/CNFs are onboarded overtime for purposes like: scalability, load balancing, availability, resilience, migration, upgrades.

It is expected that the virtualization techniques, onboarding procedures and the interfaces associated with the NFVI are not specific to the elements involved in the vOMCI solution and are not specified in this document.

5.2 vOMCI Function Description

The vOMCI function is responsible for:

- Translating a data model-based input received via the vOLT Management Function to vOMCI function interface (MVOLTMF-VOMCI), see section 5.7
- Converting the received input into a corresponding set of one (1) or more OMCI messages
- If OMCI MEs are required to be created, the vOMCI function creates the needed OMCI MEs
- Transmitting in sequence the OMCI messages to the OLT where the target ONU is attached
- If an OMCI messages error occurs, returning an error message to the vOLT Management Function for the vOLT Management Function to recover the ONU to a good state, if needed
- Translating the OMCI message (e.g., responses to requests, notification events) received from the ONU into data model-based inputs to the vOLT Management Function

In some cases, ONU management might require communication with the ONU via PLOAM (solely or in conjunction with OMCI) in order to properly serve a YANG request (a list of scenarios is provided in section 5.2.6).

PLOAM is a protocol that is implemented at the OLT for low layer functions ([2] - [5]). Thus, a requirement is that the OLT exposes APIs by which the vOMCI can exploit the PLOAM channel, directly or indirectly.

As such, the vOMCI function is further responsible for:

- Converting the received YANG input into a corresponding set of requests towards the OLT, to exchange data over the PLOAM channel.
- Guaranteeing the correct message sequence if joint PLOAM and OMCI channels should be used to serve a YANG request for which coordination of messages is required.

The translation of the data model to and from OMCI messages removes the need for the OLT and vOLT Management Function to have knowledge of the ITU-T G.988 OMCI messages formats for basic and extended managed entities with the associated sequences needed to perform the OMCI operations. In addition, the vOMCI function decouples changes in the data model (e.g., updates, extensions) of the vOLT Management Function from the OMCI messages supported by the ONU.

The interfaces provided by the vOMCI function are depicted in Figure 5.2-1.

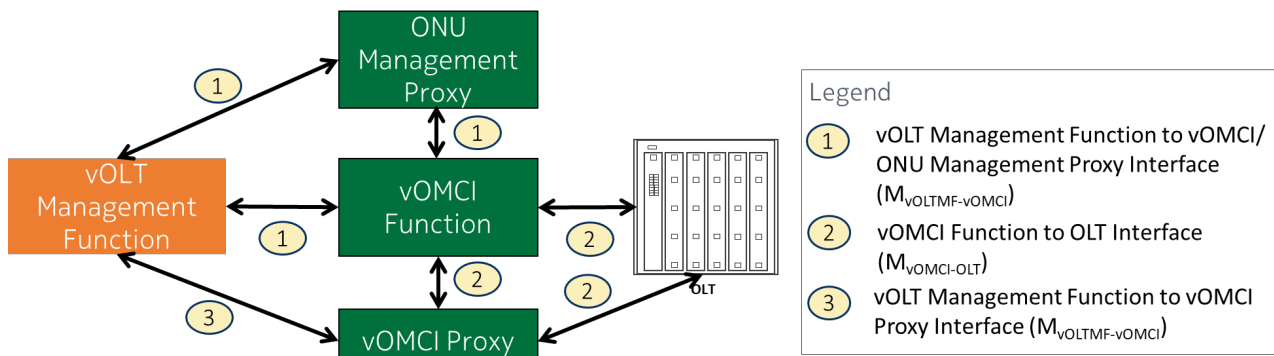


Figure 5.2-1/Figure 2 vOMCI Function and vOMCI Proxy Interfaces

The vOMCI function, ONU Management Proxy and the vOMCI Proxy are VNFs/CNFs can be integrated within and/or consumed as external microservices by the BAA layer or the CloudCO infrastructure, as described in Appendix I of this Technical Report.

5.2.1 Onboarding the vOMCI Function

The deployment of the vOMCI function as a VNF requires additional considerations for onboarding the VNF in the Service Provider's network that eases the integration of the vOMCI solution into their Cloud environment (i.e., CloudCO BAA layer, CloudCO infrastructure). Minimally this means that the vOMCI function has to be easily integrated in an Openstack or a Kubernetes infrastructure and implies that a vOMCI function requires the associated descriptors and artifacts to ease its integration in the infrastructure using well known tools such as Ansible, Heat and Helm.

5.2.2 Management of the vOMCI Function

The following management information is provided by the vOMCI function:

- State of vOMCI function attributes
- Notifications emitted by the vOMCI function
- Configuration and state of gRPC endpoints, or endpoints supported by other protocols;
- Configuration and state of the OMCI message retransmission capabilities if required by the vOMCI deployment scenario
- Statistics of OMCI message retransmission capability if required by the vOMCI deployment scenario
- Statistics of vOMCI message requests and responses
- Operational information regarding ONUs associated with the vOMCI function

The data model for the management information defined above is defined in the vOMCI function's YANG model defined in Annex A: vOMCI YANG Modules.

Additionally, inventory information for the vOMCI function is provided through the administrative interface that deployed the vOMCI function.

5.2.2.1 VNF/CNF Requirements

R-vOMCI_FUNC.10 – A vOMCI function instance **MUST** be a stand-alone entity with a unique identity. For example, the vOMCI function could be deployed in the BAA layer or northbound SDN M&C.

R-vOMCI_FUNC.20 - A vOMCI function **MUST** provide the capability to manage multiple ONUs, regardless of how many OLTs to which the ONUs attach.

R-vOMCI_FUNC.30 – A vOMCI function **MUST** be capable of being deployed in an Openstack infrastructure or a Kubernetes infrastructure.

R-vOMCI_FUNC.31 - The same vOMCI function application **SHOULD** be available in variants deployable on both an Openstack and a Kubernetes infrastructure.

R-vOMCI_FUNC.50 - The vOMCI function **MUST** be packaged with the template and artifacts needed (e.g., Heat template, Helm Chart) to ease its onboarding on the infrastructure.

R-vOMCI_FUNC.51 - The NFVI **MUST** provide a lifecycle management interface for the vOMCI function.

5.2.2.2 Management Requirements

R-vOMCI_FUNC.60 - The vOMCI function **MUST** provide a $M_{\text{VOLTMF-vOMCI}}$ management interface based on YANG for configuration and state information.

R-vOMCI_FUNC.70 - The $M_{\text{VOLTMF-vOMCI}}$ interface **MUST** support the management of the vOMCI function's notifications.

R-vOMCI_FUNC.80 - The $M_{\text{VOLTMF-vOMCI}}$ interface **MUST** support the management of the vOMCI function's configuration and state of the gRPC endpoints.

R-vOMCI_FUNC.90 - The $M_{\text{VOLTMF-vOMCI}}$ interface **MUST** support the management of the vOMCI function's inventory and state.

R-vOMCI_FUNC.100 - The M_{vOLTMF-vOMCI} interface MUST support the management of the vOMCI function's configuration, state and statistics associated with the retransmission capability, if required by the vOMCI deployment scenario.

R-vOMCI_FUNC.110 - The M_{vOLTMF-vOMCI} interface MUST support the management of the vOMCI function's statistics associated with the processed vOMCI messages.

R-vOMCI_FUNC.120 - The M_{vOLTMF-vOMCI} interface MUST support the capability to create and delete ONUs for the purposes of maintaining the ONU's local information within the vOMCI function.

R-vOMCI_FUNC.130 - The M_{vOLTMF-vOMCI} interface MUST support the capability to configure the ONU management chain information for the purposes of maintaining the ONU's local information within the vOMCI function.

R-vOMCI_FUNC.140 - When modifying the information used to identify the ONU management chain, the vOMCI function SHOULD do so without reconfiguring the ONU.

5.2.2.3 ONU Discovery and Operation

This section specifies requirements for ONU Discovery and Operation derived from Appendix V: Supporting Interactions for vOMCI Deployments.

R-vOMCI_FUNC.200 The vOMCI function MUST provide the capability to send Get MIB data sync message to the ONU as defined in ITU-T G.988 9.1.3.

R-vOMCI_FUNC.210 The vOMCI function MUST provide the capability to send MibUpload and MibUploadNext OMCI messages to the ONU as defined in [ITU-T] G.988 I.1.3.

R-vOMCI_FUNC.220 The vOMCI function MUST provide the capability to send MibReset OMCI message to the ONU as defined in ITU-T G.988 I.1.4.2.

R-vOMCI_FUNC.230 The vOMCI function MUST provide the capability to report configuration align results to the vOLTMF as defined in ITU-T G.988 I.1.3.

R-vOMCI_FUNC.240 The vOMCI function MUST provide the capability to send OMCI configuration provision messages to the ONU.

R-vOMCI_FUNC.250 The vOMCI function MUST provide the capability to send a notification to the vOLTMF that the alignment of a configured ONU has completed.

R-vOMCI_FUNC.260 The vOMCI function MUST support issuing of OMCI messages to obtain ONU device and the device's software information.

RvOMCI_FUNC.270 The vOMCI function MUST support issuing of OMCI Synchronize time action defined in section A.2.33 and A.3.33 of G.988 [1] to align the 15-min PM collection interval.

R-vOMCI_FUNC.280 The vOMCI function MUST support the capability to obtain a time of day reference from a common time source using the NTP protocol defined in RFC 5905 [8].

R-vOMCI_FUNC.290 The vOMCI function MUST align the ONU to the time of day reference obtained by the vOMCI function upon discovery and realignment.

R-vOMCI_FUNC.295 The vOMCI function MUST respect OMCI dependency on ME creation and/or configuration for the subset of ONUs managed by this vOMCI function.

Note: **R-vOMCI_FUNC.295** realizes that the ONUs should respect OMCI dependencies as defined in ITU-T G.988 section 9. However, in the scenario where the ONU cannot or doesn't conform to dependencies as defined ITU-T G.988 section 9, the vOMCI function must respect the OMCI dependency of the ONUs managed by this vOMCI function.

5.2.3 OMCI Request and Response Processing

When the vOMCI function receives a request for an ONU from the vOLTMF, the vOMCI function:

- Translates the request into one or more OMCI messages
- Formulates the response from the responses to the OMCI messages
- Correlates request/responses from the vOLTMF to the request/responses sent/received from the OLT.

Note: The vOLTMF is responsible for ensuring all requests sent to the vOMCI function do not have any dependencies on any pending request which is waiting response from the vOMCI function.

5.2.3.1 Requirements

R-vOMCI_FUNC.300 - The vOMCI function MUST process YANG requests from the vOLTMF that target the same ONU on a first-in/first-out order basis.

Note: Time consuming YANG requests typically use an asynchronous request/response/notification pattern where the response to the YANG request is acknowledged and the results are provided in a separate notification. For example, software download and diagnostic YANG requests would use the asynchronous pattern.

Note: The behavior for OMCI requests for a given ONU typically uses a serialized, synchronous request/response pattern.

R-vOMCI_FUNC.301 - The vOMCI function MUST translate YANG requests received from the vOLTMF into one or more ITU-T G.988 OMCI messages for transmittal to the targeted ONU. The OMCI messages that comprise the request are transmitted serially.

R-vOMCI_FUNC.310 - The vOMCI function MUST translate received ITU-T G.988 OMCI messages to a YANG notification or a YANG response associated with a vOLTMF request.

R-vOMCI_FUNC.320 - The vOMCI function MUST translate ITU-T G.988 OMCI baseline messages as described in section A.3 of ITU-T G.988.

R-vOMCI_FUNC.330 - The vOMCI function MUST translate ITU-T G.988 OMCI extended messages as described in section A.2 of ITU-T G.988.

R-vOMCI_FUNC.340 - The vOMCI function MUST maintain the OMCI message request life-cycle including processing of time-outs and re-transmission policies per ITU-T G.988 processing across the $M_{\text{vOMCI-OLT}}$ interface.

R-vOMCI_FUNC.350 - The YANG data encoded in the *target*, *current*, *data*, *filters* and *error* payload objects of messages used on the $M_{\text{vOLTMF-vOMCI}}$ interface, MUST be encoded as per RFC 7951 [17].

R-vOMCI_FUNC.360 – If the vOMCI function fails to complete a request from the vOLTMF, the vOMCI function must return an error response to the vOLTMF/ONU Management Proxy across the $M_{\text{vOLTMF-vOMCI}}$ interface.

R-vOMCI_FUNC.370 – The vOMCI function MUST ensure that vOMCI messages received through a remote-endpoint that are sourced from an ONU is the same remote-endpoint that is assigned to the ONU management chain for the ONU.

R-vOMCI_FUNC.380 – The vOMCI function MUST provide the capability to be configured with and maintain the ONU's management chain including the ONU's name, service endpoint or entity identification corresponding to both upstream ONU Management Proxy or vOLTMF instances and downstream vOMCI Proxy instances or OLTs.

R-vOMCI_FUNC.381 – The vOMCI function instances MUST provide the capability to forward, based on the configured ONU's management chain, an ONU's vOMCI message to either the OLT or vOMCI Proxy.

R-vOMCI_FUNC.382 – The vOMCI function instances MUST provide the capability to forward, based on the configured ONU's management chain, an ONU's response to a management request that was received from the vOLTMF or ONU Management Proxy.

5.2.3.2 Bulk OMCI Transmission

An operation on an ONU over OMCI, might require the exchange of several OMCI messages (requests and responses). As such, a translation of a vOLTMF request might result into generation of multiple OMCI messages. A vOMCI entity can decide (subject to internal logic) to send each OMCI message sequentially as an individual vOMCI message (as imposed by the OMCI transaction stop-and-wait mechanism) or instead have OMCI messages grouped within one or more vOMCI message in bulk.

The bulk transmission would be beneficial from resources perspective as it reduces the number of gRPC message exchanges. A typical (straightforward) deployment example (but certainly not the only one) is the ONU SW download (I.3.2 of ITU-T G.988 [1]). In this action, a multitude of OMCI messages is sent as part of a window, for which a single response is expected.

The special situation of SW download is that it doesn't require every OMCI message to be responded. This is not a constraint for the use of the bulk OMCI transmission. Any framework exploit is implementation specific. However, special care should be given so that the improvement at gRPC layer does not affect the OMCI protocol, as a matter of OMCI message processing, responses required, retransmissions and ordering. In other words, the intended result of the OMCI interactions should be the same, irrespective of OMCI message transmission mode (one by one or in batches),

5.2.3.2.1 Requirements

R-vOMCI_FUNC.390 - The vOMCI function MUST support transmission/reception/processing of basic vOMCI message conveying a single OMCI message and SHOULD support transmission/reception/processing of bulk vOMCI message comprising of multiple OMCI messages, as defined in section 5.8.

R-vOMCI_FUNC.391 – When bulk vOMCI message is supported, the vOMCI function MUST publish the capability upon connection establishment with peer entities, using the Hello message as defined in 5.8.2.

R-vOMCI_FUNC.392 – The vOMCI function, when constructing a bulk vOMCI message, MUST encapsulate the OMCI messages in the same sequence that would be sent in the basic OMCI transmission.

R-vOMCI_FUNC.393 – The vOMCI function, when processing a bulk vOMCI message, MUST preserve the order in which the OMCI messages are encapsulated in the bulk vOMCI message.

5.2.4 OMCI Notification Processing

The vOMCI function receives ITU-T G.988 OMCI events (e.g., AVC, Alarms) as payload for a source ONU from the OLT across the $M_{\text{vOMCI-OLT}}$ interface.

Upon reception of an OMCI event, the vOMCI function:

- Translates the ITU-T G.988 OMCI alarm into a YANG notification
- Forwards translated YANG notification to the vOLTMF
- Maintains the YANG notification life-cycle including processing of time-outs and re-transmission policies

Note: The definitions of the policies used by the vOMCI function for time-out and re-transmission of notifications is outside the scope of this Technical Report.

As a vOMCI function connects to at most one (1) vOLTMF, the vOMCI function simply forwards the transmitted YANG notification across the $M_{\text{vOLTMF-vOMCI}}$ interface to the connected vOLTMF. If the vOMCI function cannot deliver the notification, the YANG notification is discarded.

5.2.4.1 Alarm and Event Handling

When an ONU detects faults or conditions that are autonomously reported to a management function as a notification they are classified as:

- Alarms
- Attribute value change (AVC) events
- Threshold Cross Alert (TCA) events

The ONU reports Alarms and TCAs using the alarm procedures documented in section 9 MIB Description and 7.2 Fault management of ITU-T G.988 [1]. In addition, when changes occur within the ONU's state, AVC events are reported to a management function using the AVC procedures documented in section 9 MIB Description of ITU-T G.988 [1].

Within the context of this Technical Report, the procedure to translate various types of OMCI notification messages (i.e., Alarm report, AVC) to the corresponding YANG notification and the corresponding reporting of the notification from the vOMCI function to the vOLTMF is specified.

Alarm-Reporting Control (ARC) is implemented by the vOMCI function by translating the Alarm reporting control attributes associated with a specific resource as represented by the resource's YANG module in TR-385 [35] into the corresponding OMCI messages for Alarm-reporting control as defined in A.1.4.3 of ITU-T G.988 [1]. For example, the ANI resource in TR-385 [35] has attributes for ARC administration. The vOMCI function translates these YANG leaf nodes into the corresponding OMCI message(s) to configure the ANI-G ME.

5.2.4.1.1 Notification Translation and Reporting

When the vOMCI function receives an OMCI Alarm message (Alarm or TCA) from an ONU, the vOMCI function translates the OMCI Alarm message for the ME (e.g., ANI-G) into the RFC 8632 [21] YANG Alarm notification for the corresponding resource identifier (e.g., ANI) and then reports the YANG notification to the vOLTMF/ONU Management Proxy using the "notification" message defined in section 5.7.1 of this Technical Report.

When the vOMCI function receives an OMCI AVC message from an ONU, the vOMCI function logs the OMCI AVC message. The translation of the OMCI AVC message is outside the scope of this Technical Report.

If the OMCI notification message cannot be translated, the notification is still translated using the "*untranslated-omci-notification*" notification specified in Annex A: vOMCI YANG Modules.

5.2.4.1.1 Alarm Synchronization

One aspect of the OMCI alarm message is the sequence number associated with the alarm instance. The vOMCI function can use the Alarm sequence number that is assigned by the ONU. The procedure for synchronizing Alarms between the vOMCI function and the ONU is described in section A.1.4.2 of ITU-T G.988 [1].

If the vOMCI function determines that the OMCI alarm message's sequence number is not aligned with the vOMCI function's expected sequence number, the vOMCI function sends an ONU specific event notification to the vOLTMF that informs the vOLTMF that the vOMCI function and the ONU are out of alignment for Alarms.

The vOLTMF then sends a synchronize request to the vOMCI function to synchronize the alarms.

Note: The reason that the vOMCI function does not autonomously synchronize the alarm state but relies on the vOLTMF to perform the action is because in some circumstances the OLT may be a management function (e.g., ToD) for the ONU as well as the vOLTMF. As such the vOLTMF is the function that has the responsibility to ensure alarms are synchronized with the ONU as described in section 5.6.2 of this Technical Report.

5.2.4.2 Test Invocation and Result Handling

ONUs that have implemented tests using OMCI as described in section A.1.5 of ITU-T G.988 [1] can have these tests invoked by the vOLTMF using the "action" and "rpc" operations as described in section 5.7.1 of this Technical Report.

In section A.1.5 of ITU-T G.988 [1], OMCI tests are asynchronous in nature when the test is invoked by the OLT. A test response that is received from the ONU indicates that the ONU has accepted the test for processing and results of the test are then sent at a later time by the ONU to the OLT as a test result notification using the transaction identifier to correlate the test results with the test invocation.

As noted previously, the information element that is common to the Test invocation message and the Test result message is the transaction identifier that was identified by the Test invocation message. This transaction identifier is part of all ONU YANG Test RPCs and Test result notifications specified in TR-383 [33] and TR-385 [35] that are aligned with the OMCI Test result message formats specified in section A.2.39 of G.988 [1].

During the translation of the YANG test to the associated OMCI ME that implements the test, the vOMCI function is required to wait for the test results, if successful or test failure in order to reply to the "rpc" or "action" operation when the "rpc" or "action" operation requires the test results to be returned.

5.2.4.3 Requirements

R-vOMCI_FUNC.400 - The vOMCI function MUST translate OMCI alarm messages received across M_{vOMCI-OLT} interface into an RFC 8632 [21] alarm-notification to be transmitted across the M_{vOLTMF-vOMCI} interface.

R-vOMCI_FUNC.405 - The vOMCI function MUST provide the capability to forward, based on the configured ONU's management chain, the ONU's notification to the associated vOLTMF or ONU Management Proxy for which the vOMCI function has received management requests for the same ONU.

R-vOMCI_FUNC.410 - The vOMCI function MUST log OMCI attribute-value-change messages received across M_{vOMCI-OLT} interface.

R-vOMCI_FUNC.420 - If the vOMCI function cannot translate an OMCI Alarm or AVC message into a specific YANG notification, the vOMCI function MUST translate the OMCI message into a vOMCI *untranslated-omci-notification* notification defined in Annex A: vOMCI YANG Modules.

R-vOMCI_FUNC.421 - The vOMCI function MUST be capable of sending an *onu-alarm-misalignment* notification defined in Annex A: vOMCI YANG Modules whenever the vOMCI function is unable to align the vOMCI function's alarms for an ONU with the alarms reported by the ONU.

R-vOMCI_FUNC.430 - The vOMCI function MUST discard YANG notifications that cannot be transmitted across the M_{vOLTMF-vOMCI} interface.

R-vOMCI_FUNC.440 – When detected, the vOMCI function MUST support sending notifications regarding the failure of communication with the remote endpoint used for vOMCI communication (e.g., gRPC channel failure).

5.2.5 Performance Monitoring

Section I.4 of ITU-T G.988 [1] describes the processing behavior for obtaining Performance Monitoring (PM) data (i.e., current, historical) from ONUs using OMCI classical and extended PM managed entities. This section of the Technical Report describes the responsibilities the vOMCI function has in obtaining and returning performance monitoring data in the context of on-demand and periodic collection of performance data.

5.2.5.1 On-demand Retrieval of PM Data

On-demand requests for PM data from ONUs are obtained using the vOMCI function's request and response processing as described in section 5.2.3 of this Technical Report using the "get" message described in section 5.7.1 of this Technical Report. This procedure obtains the "latest" data record for the requested PM counter.

5.2.5.2 Periodic Collection of PM Data

The periodic collection of PM data has not been specified in this issue of this Technical Report. Future versions of this Technical Report will specify this functionality.

5.2.5.3 Collection of BIP Error Counters

The main logic on ONU PM collection from vOMCI function relies on the premises that ONUs maintain:

1. OMCI PM MEs to capture PM data.
2. A set of counters per ME attribute to store the current and the history values respectively.

Thus, the vOMCI function queries the respective counter from the ONU via OMCI, depending on the YANG request received from the vOLTMF.

When considering BIP errors on G-PON [2] the ONU behavior is different, though. There is a single counter at the ONU, which accumulates the detected BIP errors during the BER interval. Once the interval is expired, the value is sent to the OLT via PLOAM.

In order to align the logic between G-PON BIP PM and PM for other xPON types, for G-PON the pOLT replaces the ONU in the counter accumulation role, mandating the vOMCI function to perform the same queries to the OLT for G-PON PM as the queries it would do to the ONU via OMCI for other xPON types.

Queries to the OLT for G-PON PM make use of the service provided by the OLT as specified in section 5.5.5 and section 5.8.1.

5.2.6 PLOAM Dependencies

The vOMCI function is responsible to bridge the YANG and PON management models for ONU management. This mainly involves the translation between YANG and OMCI, however, PLOAM is another channel to manage ONUs, which must be utilized in conjunction with OMCI in several cases, to properly serve a YANG request. In particular, PLOAM should be used in any of the cases:

- A specific ONU request is served over PLOAM on certain PON technologies (instead of OMCI used on other PON technologies).
- Both OMCI and PLOAM messages are required to serve a certain YANG request.

The following subsections summarize some scenarios.

5.2.6.1 BIP Errors Counter

The bbf-xpon-performance-management [35] YANG model defines the “in-bip-errors” counter under the current and history PM interface state containers.

Configuring and obtaining the respective values from the ONU differentiates based on the PON technology as below:

- G-PON (G.984.3 11.2.2) [2]
 - Provision the accumulation interval via the “BER Interval” PLOAM message
 - Collect the counters on every BER interval expiration via the Remote error indication (REI) upstream PLOAM message
- XG-PON (G.988 9.16.3) [1]
 - Create a “TWDM channel PHY/LODS performance monitoring history data” OMCI ME
 - Query the OMCI ME “BIP-32 bit error count” attribute

5.2.6.2 T-CONT Provisioning

T-CONT provisioning is managed by the bbf-xpongenmtcont YANG model [35], by which the creation/deletion of a T-CONT to an ONU is combined with the allocation/deallocation of the corresponding Alloc-ID.

The ITU-T recommendations on OAM, for provisioning a T-CONT to an ONU requires to:

- First assign the Alloc-ID via “Assign_Alloc-ID” PLOAM message
- Then bind the Alloc-ID to a T-CONT ME (G.988 9.2.2 [1]) via OMCI message

Thus, on every PON technology, T-CONT management is a joint collaboration between PLOAM and OMCI. Moreover, in several ITU-T recommendations, the aforementioned ordering between the actions is implied.

5.2.6.3 AES Encryption

In YANG, the bbf-xpongenmtcont.yang [35] model provides the configuration leaves to enable/disable AES encryption on an ONU’s GEM port.

As per ITU-T, the following is required to enable AES for a GEM port on ONUs:

- Create GEM port (G.988 9.2.3 [1]) via OMCI
- Enable encryption on GEM port
 - G-PON: Encrypted_Port-ID message via PLOAM (G.984.3 9.2.1 [2])
 - XG-PON: Configure “Encryption key ring” on GEM port via OMCI

Hence, provisioning AES encryption on ONU is PON technology dependent and in case of G-PON a joint OMCI and PLOAM communication is required.

5.2.6.4 Requirements

R-vOMCI_FUNC.600 – The vOMCI function MUST be capable to distinguish the need for OMCI vs PLOAM interface for ONU management and utilize the appropriate channel to serve YANG requests received from the vOLTMF.

R-vOMCI_FUNC.601 – The vOMCI function MUST translate YANG requests received from the vOLTMF into one or more requests towards the OLT to trigger the OLT to transmit ITU-T PLOAM messages to the targeted ONU, and/or query the OLT for ONU state data collected via PLOAM, as decided per **R-vOMCI_FUNC.600**. The requests MUST contain all required data to construct a PLOAM message when needed.

R-vOMCI_FUNC.610 – The vOMCI function MUST translate the responses it receives from the OLT following the requests done per **R-vOMCI_FUNC.601**, to a YANG response associated with a vOLTMF request.

5.2.7 OLT – ONU Secure Mutual Authentication

ITU-T PON enhanced security recommendations (Annex B.2 of ITU-T G.984.3 [2]; Section 15.2.2 in ITU-T G.987.3 [3] and Annex C in ITU-T G.989.3 [5]; Annexes C.15.2.2, C.C of ITU-T G.9807.1 [4]; Section 9.13.11 in ITU-T G.988 [1]) describe optional mechanisms for the OLT and the ONU to mutually authenticate each other:

- Upon completion of ONU activation, the OLT initiates the secure mutual authentication process
- During the process, both the OLT and the ONU compute the secure master session key (MSK) and a set of secure shared keys
- The OLT may initiate re-authentication at any time, based on specific policies

One of the recommended methods is secure mutual authentication deployed over OMCI (SMA-OMCI), which is the focus of following subsections.

5.2.7.1 OMCI Based Secure Mutual Authentication

The process, as described in Section 9.13.11 in ITU-T G.988 [1], comprises intuitively a three-step hash-based authentication (requiring a single message for each step communicated between the peer entities):

- Message 1: (Peer 1 → Peer 2) my_cryptographic_capabilities | random_challenge_1
- Message 2: (Peer 2 → Peer 1): selected_cryptographic_capabilities | random_challenge_2 | MsgHash (PSK, (selected_cryptographic_capabilities | random_challenge_1 | random_challenge_2, peer_1_identity))
- Message 3: (Peer 1 → Peer 2): MsgHash (PSK, (selected_cryptographic_capabilities | random_challenge_2 | random_challenge_1 | peer_2_identity))

Each step's message, results into several OMCI messages, as depicted in ITU-T G.988(10) F9.13.11-1 [1]. Moreover, OMCI protocol defines AVCs which aim to notify the OLT that SMA-OMCI related data are generated and ready to be collected. Then the OLT must do so over OMCI message exchanges.

Notice that the OLT is the Authenticator (Peer 1) and the ONU is the Supplicant (Peer 2).

Abiding to the ITU-T recommendations, the vOMCI framework is a mediation component and the SMA-OMCI process can be decoupled in the following domains of interactions:

- OLT – vOMCI Function: The high-level messages described above are exchanged between the peer entities
- vOMCI Function – ONU: As with any other process, the modules interact over normal OMCI. Hence the OMCI sequences defined in ITU-T G.988(10)_F9.13.11-1 [1] must be performed

Hence, the overall procedure is illustrated in Figure 5.2-2.

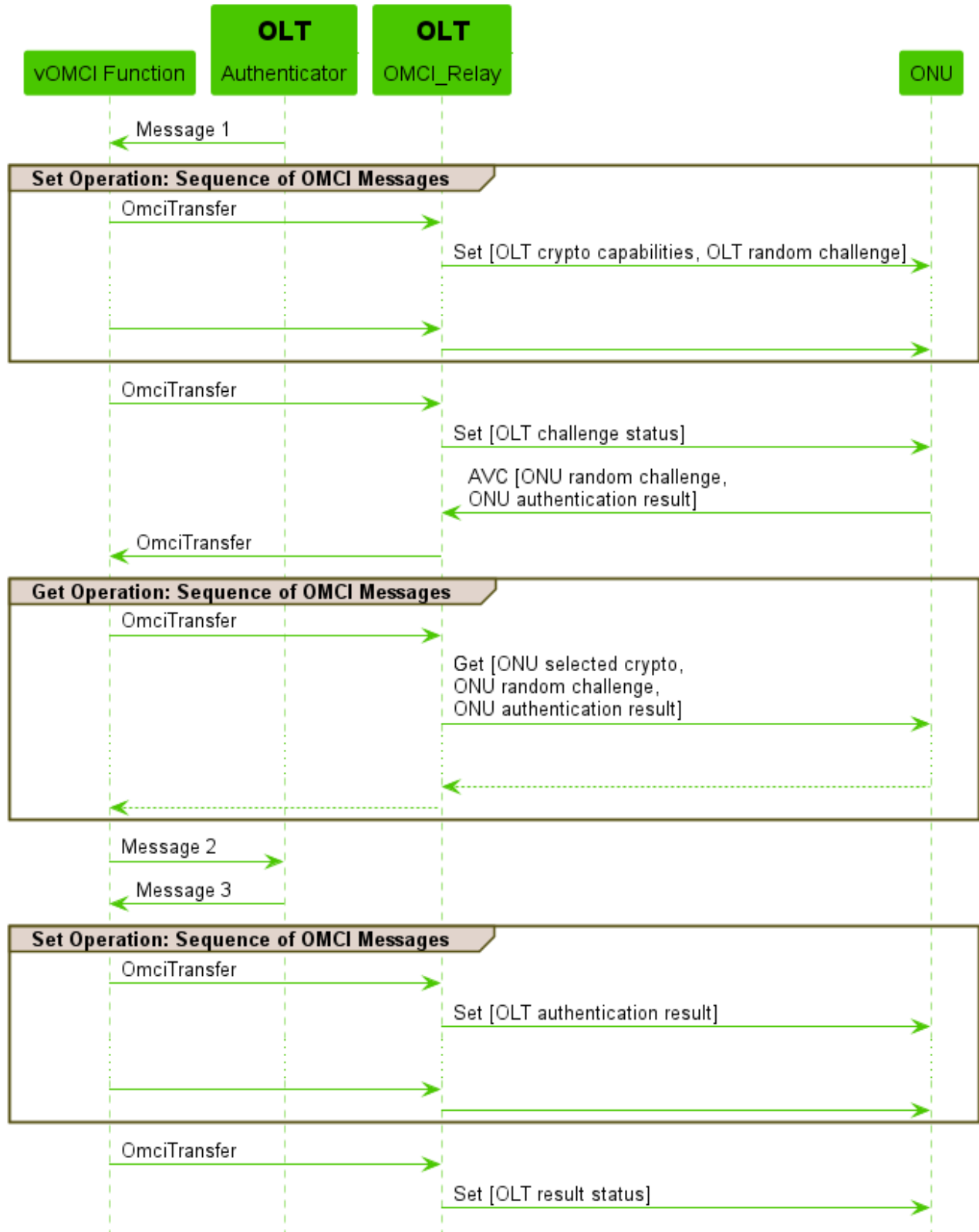


Figure 5.2-2 Secure Mutual Authentication end-to-end message flow

Within this context, the vOMCI function must support this SMA-OMCI service by

- Accepting high-level SMA-OMCI messages from the OLT as mentioned above and defined in Section 5.9
- Translate the SMA-OMCI messages received from the OLT, into a sequence of OMCI messages towards the ONU
- Transforming the OMCI messages generated by the ONU to high level SMA-OMCI messages. The task corresponds into a set of internal operations (that may result into a sequence of additional OMCI messages) required to collect ONU data and construct an SMA-OMCI message intended for the OLT

5.2.7.2 Requirements

R-vOMCI_FUNC.700 – The vOMCI function MUST translate SMA-OMCI messages received from the OLT into one or more ITU-T G.988 OMCI messages for transmittal to the targeted ONU, as defined in section 5.9.

R-vOMCI_FUNC.710 – The vOMCI function MUST translate ITU-T G.988 OMCI messages received from the ONU to a corresponding SMA-OMCI message and forward to the OLT associated with the ONU, as defined in Section 5.9.

R-vOMCI_FUNC.711 – The vOMCI function MUST be capable of autonomously collecting from the ONU, all required information to form a valid SMA-OMCI message.

R-vOMCI_FUNC.712 – The vOMCI function MUST respond with an error message as a result of failure on sending to the ONU or collecting from the ONU all information related to an SMA-OMCI message.

R-vOMCI_FUNC.713 – The vOMCI function MUST be capable of reacting on AVCs, related to SMA-OMCI, generated by the ONU and indicate that specific SMA-OMCI data are available for retrieval. The vOMCI function MUST subsequently collect required data from the ONU, through a sequence of OMCI GET messages.

5.3 ONU Management Proxy

The ONU Management Proxy is an optional entity that is introduced between vOLTMF and vOMCI function in certain vOMCI deployment scenarios to off-load the following capabilities from the vOLTMF:

- Assuming the responsibility of association of ONUs to vOMCI functions where the determination of which vOMCI function instance is used to manage an ONU, knowing and reacting to the operational state of vOMCI functions instances regarding what ONUs can be managed by which vOMCI Functions.
- Simplifying the connectivity between the vOLTMF and the vOMCI function instances where the vOLTMF only needs to connect to fewer even one ONU Management Proxy instance and not directly to all the vOMCI function instances for the ONUs the vOLTMF manages.
- Providing load balancing capabilities for ONU management traffic among vOMCI function instances.

The ONU Management Proxy is responsible for forwarding ONU management requests to vOMCI function instances from vOLTMF instances based on the association knowledge and state between ONU and vOMCI function instances.

The ONU Management Proxy correlates ONU management requests/responses between vOLTMF instance and vOMCI function instance for a specific ONU.

The ONU Management Proxy load-balances ONU associations across available vOMCI function instances based on the ONU manufacturer, software version or other information that can be associated within an ONU.

The ONU Management Proxy determines the ONU's management chain across vOLTMF instances, ONU Management Proxy instances, vOMCI function instances, vOMCI Proxy instances, and OLTs. The ONU Management Proxy can be configured with the ONU's management chain or determine the ONU's management chain as vOMCI function instances dynamically scale in or out.

Note: Only one entity (i.e., vOLTMF, ONU Management Proxy) is responsible for fulfilling the vOMCI service logic that includes the association of the ONU to the vOMCI function instance including the calculation of the ONU management chain for the association.

5.3.1 Management of the ONU Management Proxy

The following management information is provided by the ONU Management Proxy:

- State of ONU Management Proxy function attributes
- Configuration and state for the vOMCI service such as the vOMCI entities comprising the service. For vOMCI functions the vendor and software version as well as supported capabilities (e.g., Replica support) is provided.
- Configuration of the connectivity information for the links between the ONU Management Proxy, vOMCI functions, vOMCI Proxies and OLTs. The link information includes properties such as whether the link is established on demand and the type of topology (i.e., mesh, star, point-to-point). The type of topology is used by the ONU Management Proxy to determine the connectivity between the vOMCI function instance and vOMCI Proxy, ONU Management Proxy and OLTs when the vOMCI Proxy or vOMCI function instance scale in and out.
- Configuration and state of gRPC endpoints
- Statistics of ONU management requests and responses
- Operational information regarding ONUs, such as which vOMCI function instance is in charge of management of a specific ONU

The data model for the management information defined above is defined in the ONU Management Proxy's YANG model defined in Annex A: vOMCI YANG Modules.

Additionally, inventory information for the ONU Management Proxy function is provided through the administrative interface that deployed the ONU Management Proxy function.

5.3.2 General Requirements

R-ONU_MANAGEMENT_PROXY.10 – The ONU Management Proxy MAY be deployed between the vOLTMF and vOMCI functions in the vOMCI solution.

R-ONU_MANAGEMENT_PROXY.15 - The ONU Management Proxy MUST have the ability to be configured with the following information elements to be associated with an ONU:

- ONU attachment points to an OLT that includes: OLT name, xPON technology, channel termination name and PLOAM ONU-ID
- ONU name that identifies the ONU to the vOMCI function and vOMCI Proxy for requests and notifications across the MvOLTMF-vOMCI interface

- ONU characteristic information including vendor, software version

R-ONU_MANAGEMENT_PROXY.20 - When deployed, the ONU Management Proxy MUST be capable of connecting to both multiple instances of vOLTMF and vOMCI function instances using either gRPC or Kafka as defined in section 5.5 of this Technical Report according to the connectivity configuration info.

R-ONU_MANAGEMENT_PROXY.30 – When deployed, the ONU Management Proxy MUST be capable of maintaining and managing (i.e., creating, deleting, updating and querying) the association between the ONU's name and the corresponding information (e.g., vendor, software version) of the vOMCI function instance.

R-ONU_MANAGEMENT_PROXY.31 –When deployed, the ONU Management Proxy MUST be capable of determining and maintaining the ONU's management chain across vOLTMF instances, ONU management instances, vOMCI function instances, vOMCI proxy instances, and OLTs for for communication of ONU management commands and vOMCI messages, based on both ONU's info such as vendor, version info corresponding to a specific vOMCI function instance version, and the connectivity/topology across ONU Management Proxy, vOMCI function instances, vOMCI proxy, OLTs ensuring the reachability to the OLT to which the ONUs are attaching.

R-ONU_MANAGEMENT_PROXY.36 -When deployed, the ONU Management Proxy MUST have the option to be configured with the ONU's management chain.

R-ONU_MANAGEMENT_PROXY.37 -When deployed, the ONU Management Proxy MUST provide the capability to either support the configuration of the ONU management chain as defined in **R-ONU_MANAGEMENT_PROXY.36** or determine of the ONU management chain as defined in **R-ONU_MANAGEMENT_PROXY.31**.

R-ONU_MANAGEMENT_PROXY.32 – When deployed, the ONU Management Proxy MUST be capable of notifying the vOLTMF when the ONU's management chain is modified by the ONU Management Proxy. The notification allows for multiple ONUs and contains the new ONU management chain.

R-ONU_MANAGEMENT_PROXY.33 – When deployed, the ONU Management Proxy MUST be capable of associating the ONUs to the valid vOMCI function instances based on the association policy between ONU and vOMCI function instances.

R-ONU_MANAGEMENT_PROXY.34 –When deployed, the ONU Management Proxy MUST be capable of configuring the vOMCI function instances, vOMCI proxy instances with the related ONU's management chain info including the ONU's name, service endpoint info or entity identification corresponding to their both upstream and downstream neighbor entities.

R-ONU_MANAGEMENT_PROXY.35 –When deployed, the ONU Management Proxy MUST be capable of migrating the management of ONUs to other valid vOMCI function instances, updating the migrated ONUs' ONU management chain and notifying the ONU management chain dynamic change to the related vOMCI function instances, vOMCI proxy instances and vOLTMF when vOMCI function instances dynamically scale in or out.

R-ONU_MANAGEMENT_PROXY.40 – When deployed, the ONU Management Proxy MUST be capable of forwarding ONU management commands defined in section 5.7 of this Technical Report between vOLTMF and vOMCI function instances.

R-ONU_MANAGEMENT_PROXY.41 – When deployed, the ONU Management Proxy MUST be capable of forwarding a management command (e.g., setting the connectivity information that would be used by the vOMCI function instances) targeted to a type of a vOMCI function to all of the associated vOMCI function instances.

R-ONU_MANAGEMENT_PROXY.42 – When deployed, when the ONU Management Proxy does not receive a response to a command when configuring a vOMCI function instance, the ONU Management Proxy MUST be capable of retrying the command to the failed instance. After a specific number of retries, the ONU Management Proxy MUST produce a notification to the vOLTMF for the failure to receive a response. In addition, the ONU Management Proxy internally resolves the failure so that future requests are not sent to the failed instance.

R-ONU_MANAGEMENT_PROXY.43 - When deployed, the ONU Management Proxy MUST provide the capability to forward the vOMCI function instances' notifications to all of the associated vOLTMF instances.

R-ONU_MANAGEMENT_PROXY.50 - When deployed, the ONU Management Proxy MUST be capable of load-balancing the ONU management requests among multiple vOMCI function instances based in accordance with the association policy between ONU and vOMCI function instances.

R-ONU_MANAGEMENT_PROXY.51 - When deployed, the ONU Management Proxy MUST be capable of correlating ONU management requests and responses conveyed between the vOLTMF and the vOMCI functions instance for a specific ONU.

R-ONU_MANAGEMENT_PROXY.52 - When deployed, the ONU Management Proxy MUST be capable of forwarding the ONU's notifications to the associated vOLTMF that has sent management requests for the same ONU.

5.3.3 VNF/CNF Requirements

R-ONU_MANAGEMENT_PROXY.60 – An ONU Management Proxy MUST be a stand-alone entity with a unique identity.

R-ONU_MANAGEMENT_PROXY.70 –An ONU Management Proxy MUST be capable of being deployed in an Openstack infrastructure or a Kubernetes infrastructure.

R-ONU_MANAGEMENT_PROXY.71 - The same ONU Management Proxy application SHOULD be available in variants deployable on both an Openstack and a Kubernetes infrastructure.

R-ONU_MANAGEMENT_PROXY.80 - The ONU Management Proxy MUST be packaged with the template and artifacts needed (e.g., Heat template, Helm Chart) to ease its onboarding on the infrastructure.

R-ONU_MANAGEMENT_PROXY.81 - The NFVI MUST provide a lifecycle management interface for the ONU Management Proxy.

5.3.4 vOLTMF to ONU Management Proxy Management Interface ($M_{vOLTMF-vOMCI}$)

R-ONU_MANAGEMENT_PROXY.200 - The $M_{vOLTMF-vOMCI}$ interface MUST support the management of the ONU Management Proxy's configuration and state of the message channels as either gRPC or Kafka.

R-ONU_MANAGEMENT_PROXY.210 - The $M_{vOLTMF-vOMCI}$ interface MUST support the configuration and state for the vOMCI service such as the vOMCI entities comprising the service. For vOMCI functions the vendor and software version as well as supported capabilities (e.g., Replica support) is provided.

R-ONU_MANAGEMENT_PROXY.220 - The $M_{vOLTMF-vOMCI}$ interface MUST support the configuration of the connectivity information for the links between the ONU Management Proxy, vOMCI functions, vOMCI Proxies and OLTs. The link information includes properties such as whether the link is established on demand and the type of topology (i.e., mesh, star, point-to-point).

R-ONU_MANAGEMENT_PROXY.230 - The $M_{vOLTMF-vOMCI}$ interface MUST support the management of the ONU Management Proxy's inventory and state.

R-ONU_MANAGEMENT_PROXY.240 - The $M_{vOLTMF-vOMCI}$ interface MUST support the management of the ONU Management Proxy's statistics associated with the processed ONU management and control commands.

R-ONU_MANAGEMENT_PROXY.250 – When detected, the $M_{vOLTMF-vOMCI}$ interface MUST support sending notifications regarding the failure of communication with the remote endpoint used for ONU management and control commands communication (e.g., gRPC channel failure).

R-ONU_MANAGEMENT_PROXY.260 - The $M_{vOLTMF-vOMCI}$ interface MUST support the management of the vOMCI function instance's notifications for a type of vOMCI function instances based on a set of criteria (i.e., vOMCI function vendor, vOMCI function software version).

R-ONU_MANAGEMENT_PROXY.270 - The $M_{vOLTMF-vOMCI}$ interface MUST support the management of the vOMCI function's configuration and state of the gRPC endpoints for a type of vOMCI function instances based on a set of criteria (i.e., vOMCI function vendor, vOMCI function software version).

R-ONU_MANAGEMENT_PROXY.280 - The $M_{vOLTMF-vOMCI}$ interface MUST support the management of the vOMCI function's inventory and state for a type of vOMCI function instances based on a set of criteria (i.e., vOMCI function vendor, vOMCI function software version).

R-ONU_MANAGEMENT_PROXY.290 – The $M_{vOLTMF-vOMCI}$ interface MUST support the retrieval of a vOMCI function's capabilities (i.e., supported OMCI versions, Replica support) for a type of vOMCI function instances based on a set of criteria (i.e., vOMCI function vendor, vOMCI function software version).

R-ONU_MANAGEMENT_PROXY.300 - The $M_{vOLTMF-vOMCI}$ interface MUST support the management of the vOMCI function's configuration, state and statistics associated with the retransmission capability for a type of vOMCI function instances based on a set of criteria (i.e., vOMCI function vendor, vOMCI function software version), if required by the vOMCI deployment scenario.

R-ONU_MANAGEMENT_PROXY.310 - The $M_{vOLTMF-vOMCI}$ interface MUST support the management of the vOMCI function's statistics associated with the processed vOMCI messages for a type of vOMCI function instances based on a set of criteria (i.e., vOMCI function vendor, vOMCI function software version).

5.4 vOMCI Proxy

The vOMCI Proxy is an optional entity. It can be introduced in certain vOMCI deployment scenarios to offer a number of advantages that includes: scalability, load balancing of ONU management traffic, improved monitoring of ONU management chain's behavior, seamless upgrade and new deployment of vOMCI function, support of vOMCI functions deployed in different administrative domains.

The vOMCI Proxy is responsible for forwarding the OMCI messages along the vOMCI message path between the OLT and vOMCI function. The vOMCI Proxy can attach additional metadata to the OMCI messages and perform functions related to the processing (e.g., OMCI message retransmission) of the OMCI message.

The vOMCI Proxy may also perform adaptation when bulk OMCI transmission is done by the vOMCI function but is not supported by the OLT, or conversely. Consistency adaptation guides are provided in Table 5.4-1.

Table 5.4-1 Adaptation of Bulk Messages by vOMCI proxy

		OLT (Supported messages)	
		Single Packet	Bulk Packet
vOMCI Function (Generated message)	Single Packet	Default message forwarding	Support of OmciPacketBulk cannot be leveraged as it adds no value. Default message forwarding
	Bulk Packet	Translate bulk packet message to multiple single packet messages	Default message forwarding

The vOMCI Proxy correlates the vOMCI message requests and responses for a specific ONU.

The vOMCI Proxy connects to other OLTs, vOMCI functions using gRPC sessions as defined in section 5.8.2 to exchange vOMCI messages.

5.4.1 Management of the vOMCI Proxy

The following management information is provided by the vOMCI Proxy:

- State of vOMCI Proxy function attributes
- Configuration and state of gRPC endpoints
- Configuration and state of the OMCI message retransmission capabilities if required by the vOMCI deployment scenario
- Configuration of the connectivity information for the links between the vOMCI Proxys and OLTs. The link information includes properties such as whether the link is established on demand and the type of topology (i.e., mesh, star, point-to-point)
- Statistics of OMCI message retransmission capability if required by the vOMCI deployment scenario
- Statistics of vOMCI message requests and responses
- Operational information regarding ONUs associated with the vOMCI Proxy function

The data model for the management information defined above is defined in the vOMCI Proxy's YANG model defined in Annex A: vOMCI YANG Modules.

Additionally, inventory information for the vOMCI Proxy function is provided through the administrative interface that deployed the vOMCI Proxy function.

5.4.2 General Requirements

R-vOMCI_PROXY.10 – The vOMCI Proxy function MAY be deployed between the OLT and vOMCI functions in the vOMCI solution.

R-vOMCI_PROXY.20 – When deployed, the vOMCI Proxy MUST be capable of forwarding vOMCI messages defined in section 5.8.1 of this Technical Report between OLTs and vOMCI functions.

R-vOMCI_PROXY.21 – When deployed, the vOMCI Proxy MUST support transmission/reception/processing of basic vOMCI message conveying a single OMCI message and SHOULD support transmission/reception/processing of bulk vOMCI message with encapsulation of multiple OMCI messages, as defined in section 5.8.

R-vOMCI_PROXY.22 – When bulk vOMCI message is supported, the vOMCI Proxy MUST publish the capability upon connection establishment with peer entities, using the Hello message as defined in 5.8.2.

R-vOMCI_PROXY.23 – When the vOMCI Proxy supports the bulk vOMCI message interface, it MUST be capable to receive the bulk vOMCI message and forward it in the format that the receiver supports. If, forwarding requires adaptation from bulk to individual, the order of encapsulated messages in the bulk vOMCI message MUST be preserved.

R-vOMCI_PROXY.30 - When deployed, the vOMCI Proxy MUST be capable of connecting to multiple instances of OLTs and vOMCI functions according to the connectivity configuration using gRPC sessions as defined in section 5.8.2 of this Technical Report.

R-vOMCI_PROXY.40 - When deployed, the vOMCI Proxy MUST be capable of performing the OMCI message retransmission function for the management of an ONU.

R-vOMCI_PROXY.50 - When deployed, the vOMCI Proxy MUST correlate the vOMCI message requests and responses for a specific ONU.

R-vOMCI_PROXY.55 - When deployed, the vOMCI Proxy MUST ensure that vOMCI messages received through a remote-endpoint that are targeted to or sourced from an ONU is the same remote-endpoint that is assigned to the ONU's management chain.

R-vOMCI_PROXY.56 –When deployed, the vOMCI Proxy MUST provide the capability to be configured with the ONU's management chain info including the ONU's ID, service endpoints info or entity identification corresponding to both upstream vOMCI function instances and downstream OLTs.

R-vOMCI_PROXY.57 - When deployed, the vOMCI Proxy MUST provide the capability to forward, based on the configured ONU's management chain, the ONU's vOMCI message requests to the OLT to which ONUs are attaching.

R-vOMCI_PROXY.58 - When deployed, the vOMCI Proxy MUST provide the capability to forward, based on the configured ONU's management chain, the ONU's notifications to the associated vOMCI function instance from which the vOMCI Proxy has received vOMCI message requests for the same ONU.

5.4.3 VNF/CNF Requirements

R-vOMCI_PROXY.60 – A vOMCI Proxy MUST be a stand-alone entity with a unique identity.

R-vOMCI_PROXY.70 – A vOMCI Proxy MUST be capable of being deployed in an Openstack infrastructure or a Kubernetes infrastructure.

R-vOMCI_PROXY.71 - The same vOMCI Proxy application SHOULD be available in variants deployable on both an Openstack and a Kubernetes infrastructure.

R-vOMCI_PROXY.80 - The vOMCI Proxy MUST be packaged with the template and artifacts needed (e.g., Heat template, Helm Chart) to ease its onboarding on the infrastructure.

R-vOMCI_PROXY.81 - The NFVI MUST provide a lifecycle management interface for the vOMCI Proxy.

5.4.4 vOLTMF to vOMCI Proxy Management Interface

R-vOMCI_PROXY.90 - The $M_{vOLTMF-vOMCI}$ interface MUST support the management of the vOMCI Proxy's configuration and state of the gRPC endpoints.

R-vOMCI_PROXY.91 - The $M_{vOLTMF-vOMCI}$ interface MUST support the management for association between vOMCI functions/OLTs/ONUs.

R-vOMCI_PROXY.100 - The $M_{vOLTMF-vOMCI}$ interface MUST support the management of the vOMCI Proxy's inventory and state.

R-vOMCI_PROXY.105 - The $M_{vOLTMF-vOMCI}$ interface MUST support the configuration of the connectivity for the vOMCI proxy instances with both vOMCI function instances and OLTs.

R-vOMCI_PROXY.110 - The $M_{vOLTMF-vOMCI}$ interface MUST support the management of the vOMCI Proxy's configuration, state and statistics associated with the retransmission capability, if required by the vOMCI deployment scenario.

R-vOMCI_PROXY.120 - The $M_{vOLTMF-vOMCI}$ interface MUST support the management of the vOMCI Proxy's statistics associated with the processed vOMCI messages.

R-vOMCI_PROXY.130 – When detected, the $M_{vOLTMF-vOMCI}$ interface MUST support sending notifications regarding the failure of communication with the remote endpoint used for vOMCI communication (e.g., gRPC channel failure).

5.4.5 PLOAM Service Relay

The vOMCI Proxy, being part of the path between the OLT and the vOMCI function, has to forward to the OLT the messages originating from the vOMCI that are defined in R-vOMCI_FUNC.601 and has to forward to the vOMCI their responses from the OLT.

Special attention should be given in a deployment that one of the peer entities proxied (either the vOMCI function or the OLT) does not implement the respective gRPC service. This could result in a situation where one of the peer entities communicates a message over this service, but the other end cannot receive it. To avoid inconsistencies on ONU service delivery, the vOMCI Proxy should discard the received message over the PLOAM related gRPC service when the receiver does not implement it. Furthermore, an error reply should be sent to the transmitter of the message with code "UNSUPPORTED_REQUEST".

Considering for example the use cases 5.2.6.1 – 5.2.6.3, such an issue can occur only in the downstream direction, because any upstream message has a downstream request as prerequisite.

5.4.5.1 Requirements

R- vOMCI_PROXY.140 – When deployed, the vOMCI Proxy SHOULD provide the capability to forward vOMCI messages pertaining to PLOAM related service, i.e. the messages defined in R-vOMCI_FUNC.601 and their responses, when both the vOMCI Function and the OLT support the corresponding interface.

R- vOMCI_PROXY.150 – When deployed and PLOAM service is supported, the vOMCI Proxy MUST be able to respond with an “UNSUPPORTED_REQUEST” error message (as defined in 5.8.1.4), when receiving a vOMCI message related to PLOAM service but the intended receiving entity does not implement the interface.

5.4.6 SMA-OMCI Relay

When the OLT employs the SMA-OMCI authenticator function, the SMA-OMCI messages may transit through the vOMCI Proxy. Thus, SMA-OMCI should be supported also on deployments with the vOMCI Proxy.

As such, the vOMCI Proxy should be capable of forwarding SMA-OMCI messages between the main actors, the OLT and the vOMCI function.

There are two basic premises to provide this functionality:

- The vOMCI Proxy should incorporate the related services and expose the corresponding interfaces
- Both proxied sides support the related services and expose the corresponding interfaces

Provided that the vOMCI Proxy supports the SMA-OMCI interface, there might be a case where the vOMCI function does not implement the respective gRPC service. Thus, SMA-OMCI messages originated by the OLT could not be served by the associated vOMCI function, while the proxy is capable of consuming them. From OLT perspective this is transparent, as it doesn't discriminate between proxy and vOMCI function. In such case, the vOMCI Proxy should discard the received message and reply to the OLT with an error response indicating that the request is unsupported.

5.4.6.1 Requirements

R- vOMCI_PROXY.160 – When deployed, the vOMCI Proxy SHOULD be capable of forwarding SMA-OMCI messages defined in section 5.9, when both main actors (OLT, vOMCI function) support the corresponding interface.

R- vOMCI_PROXY.170 – When deployed and SMA-OMCI service is supported, the vOMCI Proxy MUST be able to respond with an “UNSUPPORTED_REQUEST” error message (as defined in 5.9), when receiving an SMA-OMCI message, but the intended receiving entity does not implement the interface.

5.5 OLT Description

The Optical Line Termination (OLT) is a physical network function that in the vOMCI solution is responsible for providing the interface to the:

- Decapsulating and processing received vOMCI messages and sending them to the target ONU using the relevant PON OMCI channel.
- Process the OMCI messages received from the ONU and encapsulating them into vOMCI messages and send them to the vOMCI function or vOMCI Proxy.
- Transmitting PLOAM notifications to the vOLT Management Function

5.5.1 vOMCI Function Connectivity

A vOMCI function instance is required to manage multiple ONU devices that have been elected to be managed by that vOMCI function. The OLT has to be capable of establishing communication sessions with multiple, different vOMCI function endpoints for managing a set of ONUs.

These relationships require an association between an ONU device and a vOMCI function instance. For example, in the following figure OLT A is connected to two (2) vOMCI functions, using two (2) vOMCI communication sessions, where these vOMCI functions manage the ONUs from vendor C and D respectively.

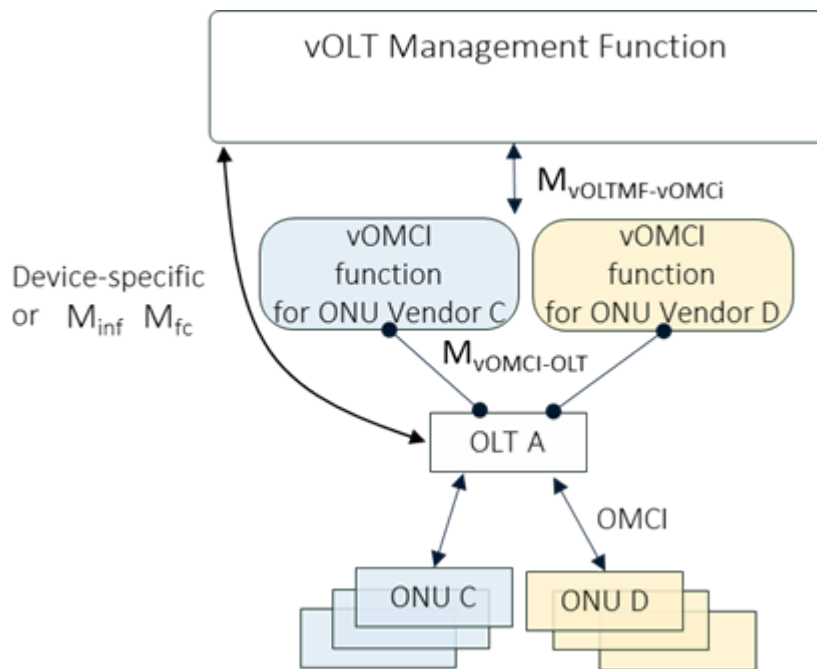


Figure 5.5-1/Figure 3 Multiple Session – OLT to vOMCI Connectivity

Likewise, the vOMCI function can establish communication sessions with multiple OLTs in order to manage one (1) or more ONUs attached to the OLT where the ONUs are associated with the vOMCI function. For example, in the following figure, the vOMCI function manages ONUs from Vendor C and D that are attached to OLT A and B respectively.

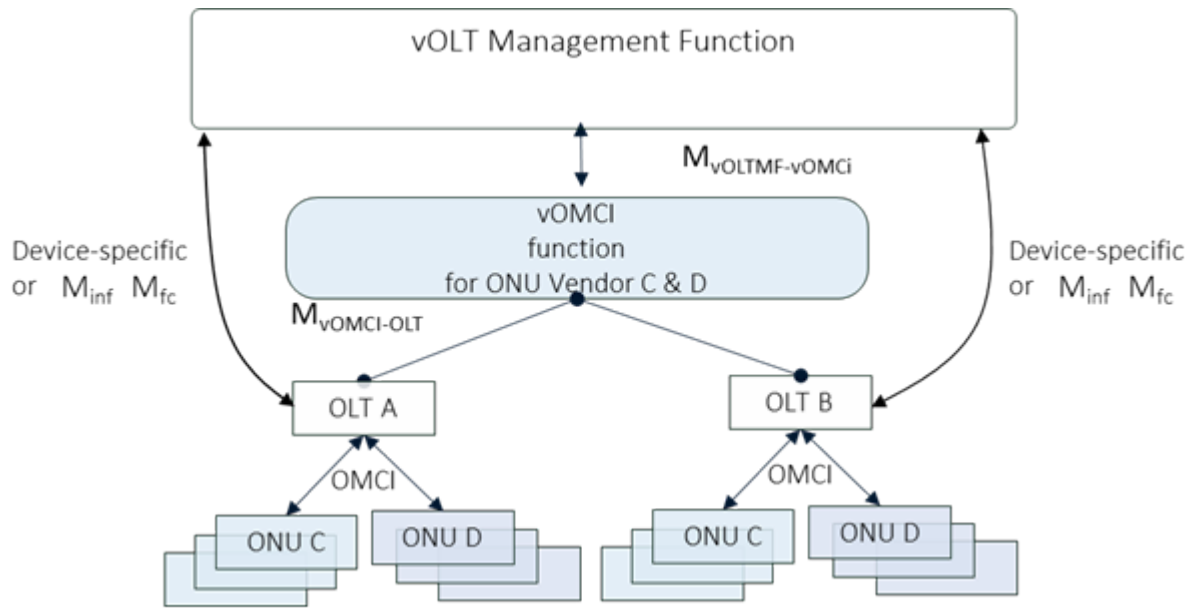


Figure 5.5-2/4 Multiple Session – vOMCI to OLT Connectivity (single vOMCI function)

In the following figure the ONUs from vendor C and D are managed by different vOMCI management functions.

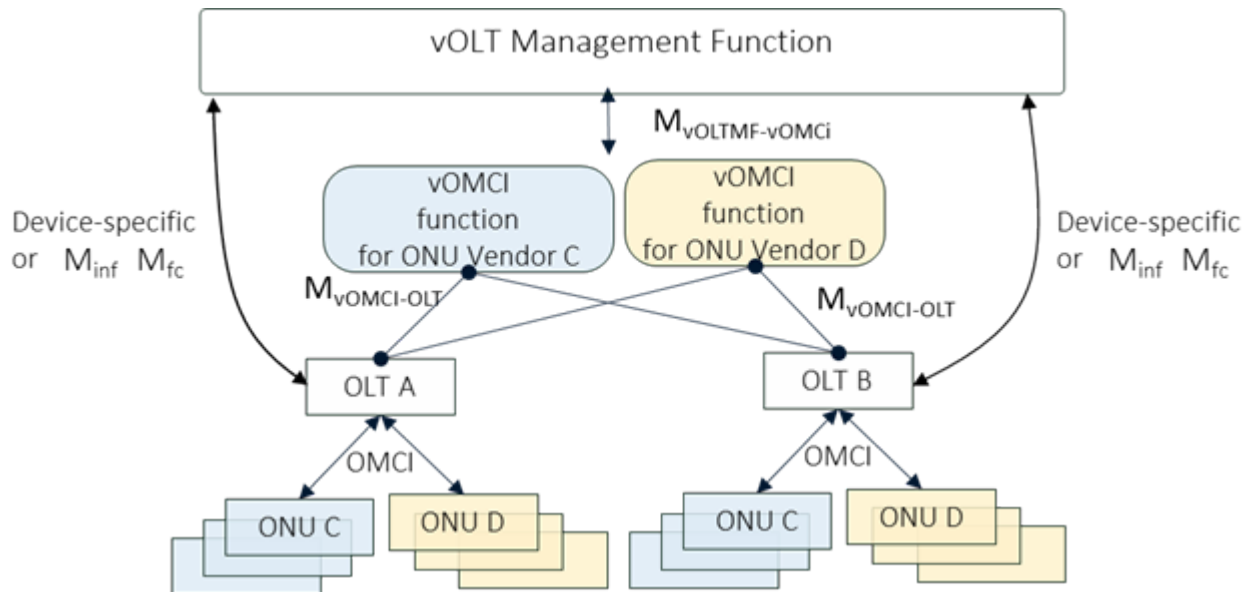


Figure 5.5-3/5 Multiple Session – vOMCI to OLT Connectivity (vendor specific vOMCI function)

In the following figure, the vOMCI Proxy is used to provide a session aggregation point toward the OLT. For example, in Figure 5.5-3 the OLT required separate communication sessions (gRPC tunnels) to vOMCI functions for ONU Vendor C and D. Using the vOMCI Proxy, OLT A would only need one communication session to the vOMCI Proxy.

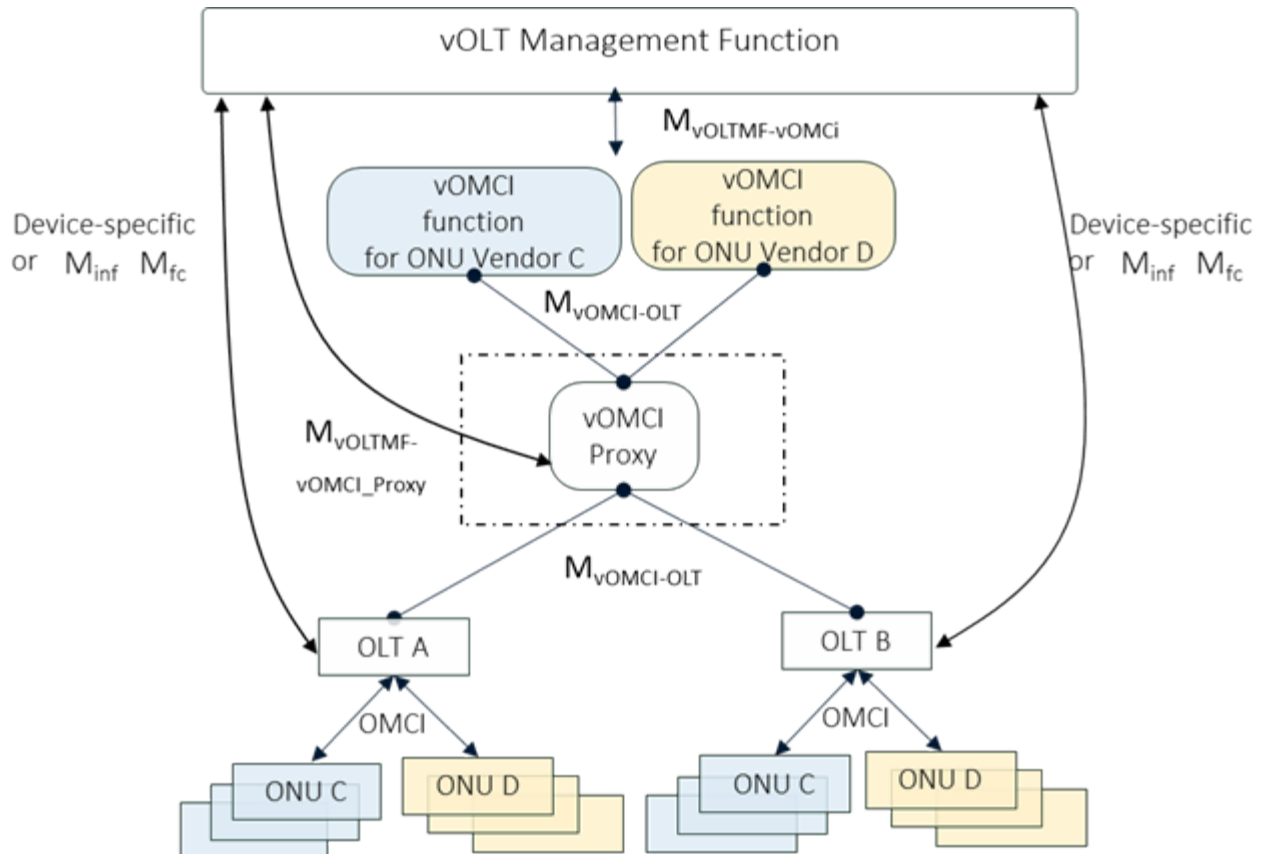


Figure 5.5-4/Figure 6 Multiple Session – vOMCI to OLT Connectivity (vOMCI Proxy)

5.5.1.1 Requirements

R-OLT.10 - The OLT MUST provide the capability to be configured with the ONU's management chain including the ONU's identifier, remote endpoints or entity identification corresponding to either upstream vOMCI function instances or vOMCI Proxy instances.

R-OLT.20 – The OLT MUST provide the capability to be configured with the ONU to remote endpoint association based on one or more of the following criteria:

- Per v-ANI as defined in TR-385 [35]
- ONU vendor
- Any ONU

R-OLT.30 - The OLT MUST provide the capability to communicate with a minimum of four (4) remote endpoints.

R-OLT.40 - The vOMCI function MUST provide the capability to communicate with one (1) or more OLTs.

R-OLT.50 - The OLT MUST be capable of processing the Network Wide ONU identifiers (i.e., OLT Id/Channel Termination Id/ONU Id) attached as additional metadata to the received OMCI messages in order to forward them on the correct OMCI channel of the target ONU.

R-OLT.60 - The OLT MUST be capable of attaching the Network Wide ONU identifier (i.e., OLT Id/ Channel Termination Id /ONU Id with the associated vOMCI function) as additional metadata to the OMCI messages

responses sent by the ONU before forwarding them to the associated remote endpoint (i.e., vOMCI function, vOMCI Proxy)

5.5.1.1.1 Activation Requirements

This section of the Technical Report provides requirements derived from Appendix V: Supporting Interactions for vOMCI Deployments for ONU activation.

R-OLT.100 - If the OLT provides G-PON interfaces, the OLT MUST support the ONU activation process as defined in Annex D.4 of ITU-T G.984.3.

R-OLT.110 - If the OLT provides XG-PON interfaces, the OLT MUST support the ONU activation process as defined in Section 12 of ITU-T G.987.3 [3].

R-OLT.120 If the OLT provides XGS-PON interfaces, the OLT MUST support the ONU activation process as defined in Annex C.12 of ITU-T G.9807.1 [4].

R-OLT.130 - If the OLT provides NG-PON2 interfaces, the OLT MUST support the ONU activation process as defined in Section 12 of ITU-T G.989.3 [5].

R-OLT.140 - If the OLT provides G-PON, XG-PON, XGS-PON or NG-PON2 interfaces, the OLT MUST send ONU presence notification message (including SN and optional PLOAM password/Registration-id) to vOLTMF as defined in BBF TR-385 issue2 and ITU-T G.984.3 [2] Appendix VI.

5.5.2 OMCI Message Forwarding

The OLT receives requests from the vOMCI function or vOMCI Proxy and processes the vOMCI message and forwards the decapsulated OMCI message to the destination ONU device through the OMCC channel that is associated with that ONU device. If the request is bulk vOMCI message, the decapsulated OMCI messages should be sent with the order in the request preserved.

Likewise, OMCI response messages and OMCI notifications received from an ONU device are processed and encapsulated into a vOMCI message (or bulk vOMCI message) and sent to the remote endpoint that is associated with that ONU device.

5.5.2.1 ToD Message Forwarding

In OMCI solutions, the OLT synchronizes the Time of Day with the ONU via the OMCI channel using the OLT-G ME defined in ITU-T G.984.3 clause 10.4.6, ITU-T G.987.3 clause 13.2, ITU-T G.989.3 clause 13.2 and ITU-T G.988 clause 9.12.2. When using the vOMCI solution, time sensitive information (i.e., GEM superframe, TstampN) sourced by the OLT cannot be provided by the vOMCI function when the vOMCI function sends the OLT-G ME, the time sensitive information has to be provided by the OLT. To permit the OLT to provide the needed time sensitive information, the vOMCI function can either use the:

- *tod_packet_msg* defined in section 5.8.1 to inform the OLT that the OLT is required to include the time sensitive information into the OLT-G ME sent by the vOMCI function
- *olt_onu_info_req* defined in section 5.8.1 to query the OLT for information needed for OMCI functioning. The response to this information includes the time sensitive information that the vOMCI function needs to pass in the OLT-G ME.

5.5.2.2 Requirements

R-OLT.200 - The OLT MUST forward the OMCI messages encapsulated in the payload of the vOMCI request that is received via the vOMCI function or vOMCI Proxy instance to the OMCC channel associated with the destination ONU device.

R-OLT.201 – The OLT MUST support transmission/reception/processing of basic vOMCI message conveying a single OMCI message and SHOULD support transmission/reception/processing of bulk vOMCI message with encapsulation of multiple OMCI messages, as defined in section 5.8.1

R-OLT.202 – When bulk vOMCI message is supported, the OLT MUST publish the capability upon connection establishment with peer entities, using the Hello message as defined in 5.8.2.

R-OLT.203 – When the OLT supports the bulk vOMCI message interface, it MUST preserve the order by which the messages are placed in the bulk vOMCI message when forwarding the individual OMCI messages to the peer entity.

R-OLT.210 - The OLT MUST calculate the Message Integrity Check (MIC) for the OMCI message and insert the MIC field into the OMCI message as defined in section 11.2.8 of ITU-T G.988 [1] prior to forwarding the OMCI message to the OMCC channel associated with the destination ONU instance.

R-OLT.220 - The OLT MUST encapsulate the OMCI messages received from the OMCC channel in the payload of a vOMCI response/notification and forward it to the vOMCI function instance that is associated with the source ONU device.

R-OLT.230 - The OLT MUST apply the GEM superframe and TstampN sourced from the OLT in the OLT-G ME when the OLT receives a *VomciMessage* message with the *tod_pack_msg* field prior to forwarding the OMCI message to the OMCC channel associated with the destination ONU instance.

R-OLT.240 – When the OLT receives a *OltMessage* message with the *olt_onu_info_req* field, the OLT MUST respond to the messages with an *olt_onu_info_resp* message.

R-OLT.250 – When the vOMCI remote-endpoint is assigned to the ONU's vANI and the OLT receives a vOMCI message targeted for an ONU, the OLT MUST ensure the remote-endpoint through which the message is received is the same remote-endpoint assigned to the vANI.

R-OLT.260 - The OLT MUST provide the capability to forward, based on the configured ONU's management chain, the ONU's vOMCI message responses to the vOMCI function instance or vOMCI Proxy instance.

5.5.3 vOMCI ONU Notifications

The OLT is the endpoint of the PLOAM channel with an ONU device where events occurring during the ONU discovery, state change, activation and signal loss that are needed by the vOLT Management function are reported by the OLT to the vOLT Management Function as notifications.

5.5.3.1 Requirements

R-OLT.300 - The OLT MUST notify the vOLTMF whenever the ONU enters the ITU-T G.984.3, ITU-T G.987, ITU-T G.9807 and ITU-T G.989 ONU Activation Operation State "O5".

R-OLT.310 - The OLT MUST notify the vOLTMF when the ONU exits the ITU-T G.984.3, ITU-T G.987, ITU-T G.9807 and ITU-T G.989 ONU Activation Operation State "O5" or POPUP state "O6".

R-OLT.320 - The OLT MUST notify the vOLTMF whenever the OLT detects or clears a Loss of OMCI Channel (LOOC) communication as defined in section 14.2.1 of G.984.3 [2].

R-OLT.330 – The OLT MUST support sending notifications for ONU presence state changes with the parameters to identify the ONU to the vOLTMF. The notification SHOULD follow the definition of ONU state change notifications defined in TR-385 [35].

R-OLT.340 – When detected, the OLT MUST support sending notifications regarding the failure of communication with the remote endpoint used for vOMCI communication (e.g., gRPC channel failure).

R-OLT.350 - The OLT MUST provide the capability to forward, based on the configured ONU's management chain, the ONU's vOMCI notifications to the associated vOMCI function instance or vOMCI proxy instance from which the OLT received the vOMCI message requests for the same ONU.

5.5.4 Management of the OLT

OLT management for the vOMCI solution uses the OLT's existing management interfaces and extends the management information to include:

- Configuration of the connectivity information for the links between vOMCI Proxys and OLTs. The link information includes properties such as whether the link is established on demand and the type of topology (i.e., mesh, star, point-to-point)
- Notifications emitted by the OLT
- Configuration and state of gRPC endpoints
- Statistics of vOMCI message requests and responses
- State of ONUs managed by the OLT

The data model for the management information defined above is defined in the pOLTs PON YANG model TR-385i2 [35] and the OLT's vOMCI YANG model identified in Annex A:vOMCI YANG Modules.

5.5.4.1 Requirements

R-OLT.400 - The OLT MUST support the configuration of the connectivity to either vOMCI function instances or vOMCI Proxy instances.

R-OLT.405 - The OLT MUST provide the capability for management of the notifications emitted by the OLT.

R-OLT.410 - The OLT MUST provide the capability for management of the configuration and state of the gRPC endpoints.

R-OLT.420 - The OLT MUST provide the capability for management of the statistics associated with the vOMCI messages processed within the OLT.

5.5.5 Expose PLOAM Service

As described in section 5.2, some ONU services management require PLOAM channel utilization.

The OLT implements the PLOAM protocol, through which several critical PON operations are performed (e.g., ONU ranging). To facilitate ONU management operations, the OLT should provide an interface by which the vOMCI function can request the OLT a PLOAM service. As such, PLOAM services can be

triggered from remote modules, while maintaining the low layer mechanism implementation and details internal to the OLT.

In particular, the OLT can receive a request from the vOMCI function that requires interaction with the ONU via PLOAM (notice that this is a separate service than the OMCI message transmission). The OLT then generates appropriate PLOAM messages and applies PLOAM mechanism to communicate with the ONU and receive potential response. When such a response from the ONU is received or when an unsolicited ONU PLOAM message is received that is applicable to the PLOAM service offered to the vOMCI function, the OLT transfers the PLOAM information (or information derived from) to vOMCI using messages defined in section 5.8.1.

5.5.5.1 BIP Accumulation

A special need of abstracting PLOAM protocol particularities from vOMCI function pertains to the G-PON BIP counters. As described in section 5.2.5, enabling BIP monitoring and collecting the values in G-PON is achieved over PLOAM and is based on push method rather than poll as opposed to OMCI PM. Moreover, the G-PON BIP does not distinguish between current and history views of the counter.

In order to align the system behavior from vOMCI function perspective, the OLT should provide the logic to control BIP counter collection from the ONU via PLOAM and accumulate the values over a period of 15-minute. Specifically, when the vOMCI function requests from the OLT the PLOAM service of enabling BIP counter at the ONU, the OLT should provision a fairly low BER interval that could address current value retrieval. Simultaneously, the OLT should instantiate a 15-minute history counter (corresponds to the latest completed 15-minute period) and a current counter (accumulates the reports from the ONU).

When afterwards, upon BER interval expiration, the ONU reports the BIP counter value, the OLT updates the current counter with the accumulated result.

When the 15-minute interval expires, the OLT will replace the 15-minute counter and restart the current counter.

The PM counter maintained by the OLT is available to vOMCI through the PLOAM Service interface.

5.5.5.2 Requirements

R-OLT.500 – The OLT SHOULD provide to the vOMCI function the capability to exploit the PLOAM protocol that the OLT implements natively (as per G.984.3 [2] section 9, G.987.3 [3] section 11), if the vOMCI function needs in order to complete a YANG service execution (this capability offered by the OLT to the vOMCI is called in short “PLOAM service”).

R-OLT.510 – When the OLT supports and enables the PLOAM service defined in R-OLT-500, it MUST translate requests received from the vOMCI function into one or more ITU-T PLOAM messages for transmittal to the targeted ONU.

R-OLT.520 – When the OLT supports and enables the PLOAM service defined in R-OLT-500, it MUST convey the POAM information received from an ONU to the vOMCI function using the messages defined in section 5.8.1, when they are associated with a service for which vOMCI function is responsible.

R-OLT.530 – When the OLT supports and enables the PLOAM service defined in R-OLT-500 and vOMCI function requests to enable BIP counter, the OLT MUST provision the BER interval to the ONU. The value should configure a frequency low enough so that it does not overload the network.

R-OLT.540 – When the OLT supports and enables the PLOAM service defined in R-OLT-500 and vOMCI function has requested to enable BIP counter, the OLT SHOULD collect and accumulate the reported values from the ONU.

R-OLT.550 – When the OLT supports and enables the PLOAM service defined in R-OLT-500 and vOMCI function has requested to enable BIP counter, the OLT SHOULD be capable to be queried for the current and the history BIP values.

5.5.6 Secure Mutual Authentication

As mentioned in section 5.2.7, the OLT, deploying an authentication function, can initiate the authentication process (based on rules and policies outside of scope of this document) for the OLT and ONU to mutually authenticate each other.

To support this functionality over vOMCI framework, the OLT should comply with the high-level interactions with the vOMCI function as defined in section 5.2.7.

The SMA-OMCI is considered an extra step for enhanced security over PON, which is employed after the ONU ranging and activation procedures are completed successfully. This means that the ONU is already allocated to a proper vOMCI function, and the OLT is able to perform OMCI interactions with the ONU.

5.5.6.1 Requirements

R-OLT.600 – When the OLT requires SMA-OMCI, it MUST be capable to generate respective high-level requests defined in section 5.9 and transmit them to the vOMCI function over the gRPC service defined in 5.9.

R-OLT.601 – When the OLT requires SMA-OMCI, it MUST be capable to process/consume respective high-level requests defined in section 5.9, received from the vOMCI function over the gRPC service defined in 5.9.

R-OLT.602 – When the OLT receives an “UNSUPPORTED_REQUEST” error message, as a response to an SMA-OMCI request, it MUST be capable to inform the SDN M&C for this specific reason of SMA failure.

5.6 vOLT Management Function Description

The vOLT Management Function (vOLTMF) is the entity that provides management of the functions associated with an OLT and the OLT's corresponding Passive Optical Networks (PON). A portion of these management functions includes the management of the ONU's attached to a OLT's PON. This section of the Technical Report describes the capabilities needed by the vOLTMF for the implementation of the vOMCI solution that includes:

- vOMCI function/vOMCI Proxy function configuration (OMCI message retransmission) and monitoring
- OLT/vOMCI function/vOMCI Proxy function infrastructure connectivity to support ONU management chains. Functionality includes creation and deletion of connections between the entities, monitoring of the connection
- Configuration of the ONU management chain using the connectivity infrastructure between the:
 - OLT and ONU Management Proxy when the ONU Management Proxy is deployed
 - OLT/vOMCI function/vOMCI Proxy function when connected directly to a vOMCI function instance
- Configuration of the connectivity information for the links between the ONU Management Proxy, vOMCI functions, vOMCI Proxys and OLTs. The link information includes properties such as whether the link is established on demand and the type of topology (i.e., mesh, star, point-to-point)
- Support for vOMCI interactions, including cancelling and rollback of interactions, between entities (e.g., OLT, vOMCI function, vOMCI Proxy, vOLTMF) to fulfill services associated with the ONU (e.g., automated ONU discovery, ONU detection, multiple and parallel operation for multiple ONUs)
- ONU configuration and operation

5.6.1 Requirements

5.6.1.1 ONU Attachment and Initial Provisioning Requirements

This section of the Technical Report defines requirements for when the vOLTMF is notified about ONU attachments along with the associated provisioning actions that happen when an ONU is powered on/off, attached or detaches from an OLT or rebooted. The requirements are derived from Appendix V: Supporting Interactions for vOMCI Deployments.

R-vOLTMF.100 - The vOLTMF MUST inform the vOMCI function, vOMCI Proxy and OLT when an ONU is to be managed using the vOMCI solution.

R-vOLTMF.110 - The vOLTMF MUST inform the vOMCI function, vOMCI Proxy and OLT when an ONU is no longer to be managed using the vOMCI solution.

R-vOLTMF.111 - The vOLTMF MUST inform the vOMCI function when an ONU is determined to be online and to be managed or has gone temporarily offline (e.g., reboot).

R-vOLTMF.112 – When ONU is determined to be online and to be managed, the vOLTMF MUST ensure the OLT and ONU are properly configured with the ONU's configuration.

R-vOLTMF.120 - The vOLTMF MUST support sending ReplaceConfig message to the vOMCI function.

R-vOLTMF.130 - The vOLTMF MUST support authentication of an ONU as defined in ITU-T G.984.3 Appendix VI, ITU-T G.987.3 section 15.2, ITU-T G.989.3 section 15.2, ITU-T G.9807.1 Annex C.15.

R-vOLTMF.140 - When an ONU is considered as a nomadic ONU that can change the OLT attachment point to which it attaches, the vOLTMF SHOULD support both authentication of an ONU as defined in ITU-T G.984.3 Appendix VI, ITU-T G.987.3 section 15.2, ITU-T G.989.3 section 15.2, and ITU-T G.9807.1 Annex C.15 and also provide capability to check if the ONU is permitted to attach to the attachment point.

R-vOLTMF.150 - The vOLTMF MUST support [R-150], [R-151], [R-154] and [R-155] in TR-156 [29] section 7.2 Initial Provisioning of ONUs.

R-vOLTMF.160 - The vOLTMF MUST support sending commands to the OLT to configure OLT resources as defined in section 5 of TR-413 [37].

R-vOLTMF.170 - The vOLTMF MUST support sending an ONU discovery notification with the discovered device and software information and the discovery status (e.g., successful, failed) to the northbound SDN M&C.

R-vOLTMF.180 - The vOLTMF MUST receive the configuration requests for PON devices from a northbound SDN M&C, which includes the identification of the PON device (i.e., OLT name, ONU ID).

R-vOLTMF.190 - If there is more than one vOLTMF instance (i.e., vOLTMF cluster) to manage PON devices, the management and control requests MUST be assigned to one of the instances based on the relationship between the PON device ID and the vOLTMF instances.

R-vOLTMF.191 - The vOLTMF instance MUST ensure that simultaneous uncommitted requests targeted to the same ONU do not have inter-dependencies between requests.

R-vOLTMF.200 - The vOLTMF MUST be the authoritative source for sending the corresponding management and control information to either OLTs or ONUs based on the identification of the PON device.

R-vOLTMF.210 - The vOLTMF MUST receive the requests for configuration and operational data of ONUs from northbound SDN M&C functions.

R-vOLTMF.220 – The vOLTMF MUST provide the capability to send notifications (e.g., device online/offline, alarms) that originate from PON devices to northbound SDN M&C functions.

R-vOLTMF.221 – When the vOLTMF does not receive a response to a command when configuring a vOMCI function instance, the vOLTMF MUST be capable of retrying the command to the failed instance. After a specific number of retries, the vOLTMF MUST produce a notification to the northbound SDN M&C function for the failure to receive a response. In addition, the vOLTMF internally resolves the failure so that future requests are not sent to the failed instance.

5.6.1.2 vOLTMF – vOMCI function Connectivity Requirements

This section of the Technical Report defines requirements for interactions between the vOLTMF and vOMCI function that were derived from Appendix VI: Connectivity Management among vOLTMF(s) and vOMCI functions.

R-vOLTMF.300 - The vOLTMF SHOULD be implemented with one or clustered instances, where each vOLTMF instance could interact with one or more vOMCI function instances directly or indirectly. For example, the ONU Management Proxy.

R-vOLTMF.310 - The vOLTMF SHOULD support parallel configuration requests of multiple different ONUs that are attached to the same or different PON links.

R-vOLTMF.320 - The vOLTMF MUST maintain the management and control states for all the ONUs.

R-vOLTMF.330 – The vOLTMF MUST provide the capability to configure the ITU-T G.988 timer expiration timeout ($T_{\max i}$) and number of retries retry counter ($R_{\max i}$) values for a given vOMCI function.

5.6.2 ONU Alarm Management

The vOLTMF provides a digital representation of the ONU's alarm instances as well as the management functionality associated with maintaining the Alarms for an ONU.

The vOLTMF represents and manages the alarm instances of an ONU by implementing the following features and resources and minimally uses the Alarm Management YANG data module as defined by RFC 8632 [21]:

- Alarm List: This feature maintains the current list of alarms
- Alarm Shelving: This feature maintains the current list of shelved alarms. A shelved alarm is an alarm that has its OMCI ARC attribute enabled.

Note: vOLTMF can implement other features of the Alarm Management YANG data module but the Alarm List resource and Alarm Shelving are the needed resources and features to interact with an ONU via the vOMCI function. As such the additional capabilities related to Alarms that can be provided by the vOLTMF is outside the scope of this Technical Report.

5.6.2.1 Requirements

R-vOLTMF.400 - The vOLTMF MUST provide the capability to retrieve the list of current alarms for an ONU.

R-vOLTMF.410 - The vOLTMF MUST provide the capability to retrieve the list of current shelved alarms.

R-vOLTMF.420 - The vOLTMF MUST provide the ability to enable and disable ONU alarm reporting.

5.6.3 ONU OMCI Topology Management

The vOLTMF maintains the topology of ONUs that includes information that identifies the ONU to the OLT, vOMCI function, vOMCI Proxy function as well as information about the OLT attachment point that connects the ONU. The topology information is necessary to:

- Maintain configuration information for the ONU that corresponds whether the ONU is managed through the vOMCI solution or is managed as an embedded solution within the OLT.
- Maintain the ONU management chain that depicts the path vOMCI messages will traverse through the OLT and if necessary, the instances of the vOMCI function and vOMCI Proxy function in order to send commands and receives responses and notifications. The ONU management chain is maintained when the ONU is managed via through the vOMCI solution.
- Maintain the association of the ONU identifiers that correspond to an ONU that are needed by the OLT, vOMCI function and vOMCI Proxy function in order to send requests and receive notifications from the entities related to the ONU.
- Inform the OLT, ONU Management Proxy and if necessary, the vOMCI and vOMCI Proxy functions when an ONU is available for OMCI communication and when the ONU is no longer available for OMCI communication. The vOLTMF uses onu-presence-state-change notifications from the OLT to help determine if the ONU has reached a onu-presence-state in which it can be communicated with through the OLT's OMCI channel (i.e., ITU-T G.984.3 Operation state O5/O6).

5.6.3.1 Requirements

R-vOLTMF.500 - The vOLTMF MUST be able to assess whether the vOMCI function can communicate with the ONU across the ONU management chain and OMCI communication channel.

R-vOLTMF.510 - The vOLTMF MUST provide the capability to receive notifications from the OLT regarding the OMCI communication channel state between the ONU and OLT.

R-vOLTMF.520 - The vOLTMF MUST provide the capability to receive notifications from the ONU Management Proxy, vOMCI function, vOMCI Proxy and OLT regarding the operational state of an ONU's management chain.

R-vOLTMF.530 – When deployed without an ONU Management Proxy, the vOLTMF MUST provide the capability to configure the vOMCI function whenever an ONU's ONU management chain is modified or the OMCI communication channel between the ONU and OLT changes.

R-vOLTMF.531 – When deployed without an ONU Management Proxy, the vOLTMF MUST provide the capability to configure the vOMCI Proxy whenever an ONU's ONU management chain is modified or the OMCI communication channel between the ONU and OLT changes.

R-vOLTMF.532 – The vOLTMF MUST provide the capability to configure the OLT whenever an ONU's ONU management chain is modified or the OMCI communication channel between the ONU and OLT changes.

R-vOLTMF.533 – When deployed with the ONU Management Proxy, the vOLTMF MUST provide the capability to configure the ONU Management Proxy whenever the OMCI communication channel between the ONU and OLT changes.

R-vOLTMF.542 - The vOLTMF MUST maintain the following information elements to be associated with an ONU:

- ONU attachment points to an OLT that includes: OLT name, xPON technology, channel termination name and PLOAM ONU-ID
- ONU name that identifies the ONU to the vOMCI function and vOMCI Proxy for requests and notifications across the M_{vOLTMF-vOMCI} interface
- ONU characteristic information including ONU vendor and software version
- ONU management chain that identifies the OLT or when deployed without an ONU Management Proxy, the path between the OLT, vOMCI Proxy, vOMCI functions and OLT for communication of vOMCI messages between the vOMCI function and the OLT.

R-vOLTMF.543 - The vOLTMF MUST provide the capability to be configured with the mapping information to associate the ONU characteristics (i.e., vendor, software info) with the versions supported by vOMCI function instances.

R-vOLTMF.550 – The vOLTMF MUST provide the ability to synchronize the ONU topology and associated ONU information elements maintained by OLT, ONU Management Proxy, vOMCI function, vOMCI Proxy and the vOLTMF. The vOLTMF is considered the authoritative source of the ONU topology information.

5.6.3.2 vOMCI Entity Connectivity Management

The vOLTMF establishes an ONU management chain for an ONU using vOMCI entities. For example, the OLT and when the ONU Management Proxy is not deployed the vOMCI function and vOMCI Proxy function. However, before the vOLTMF establishes the ONU management chain for an ONU, connectivity between the vOMCI entities must first be established. The vOLTMF maintains the connectivity information between vOMCI entities. This information includes the network addressing information and security credentials needed to establish the communication channels between vOMCI entities across the M_{vOMCI-OLT} interface.

When the ONU Management Proxy is deployed, the ONU Management Proxy establishes an ONU management chain for an ONU using vOMCI entities (i.e., vOMCI Proxy function, vOMCI function). At the same time, the ONU Management Proxy maintains the connectivity configuration information between vOMCI entities.

5.6.3.2.1 Requirements

R-vOLTMF.600 - The vOLTMF MUST maintain the information that is needed to establish the communication channel between vOMCI entities (i.e., OLT, vOMCI function, vOMCI Proxy).

R-vOLTMF.610 - The vOLTMF MUST provide the capability to configure the OLT with the connectivity information that is needed to establish the communication channel across the M_{vOMCI-OLT} interface.

R-vOLTMF.611 –When the ONU Management Proxy is deployed, the vOLTMF MUST provide the capability to be notified when an ONU's manage chain is modified by the ONU Management Proxy.

R-vOLTMF.612 –The vOLTMF MUST provide the capability to configure the OLT with the ONU to remote endpoint association defined in R-OLT.20 when there is an update to the association.

R-vOLTMF.615 - The vOLTMF MUST provide the capability to configure the ONU Management Proxy with the connectivity information for the links between the ONU Management Proxy, vOMCI functions, vOMCI Proxys and OLTs to establish the communication channel across the $M_{VOLTMF-VOMCI}$ interface. The link information includes properties such as whether the link is established on demand and the type of topology (i.e., mesh, star, point-to-point).

R-vOLTMF.620 - The vOLTMF MUST provide the capability to configure the vOMCI function with the connectivity information that is needed to establish the communication channel across the $M_{VOMCI-OLT}$ interface.

R-vOLTMF.630 - The vOLTMF MUST provide the capability to configure the vOMCI Proxy function with the connectivity information that is needed to establish the communication channel across the $M_{VOMCI-OLT}$ interface.

R-vOLTMF.640 – The vOLTMF SHOULD provide the capability to be configured by a northbound SDN M&C with the connectivity information that is needed to establish the communication channel across the $M_{VOMCI-OLT}$ interface.

5.6.4 ONU Persistent Configuration and State Management

The vOLTMF provides a digital representation of the ONU for use by various northbound SDN M&C systems. As such the vOLTMF maintains the authoritative configuration state of the ONU to which the ONU is synchronized. In addition, the vOLTMF maintains the current set of alarms for the ONU. When the vOLTMF identifies that the ONU configuration is not aligned with the configuration maintained by the vOLTMF, the vOLTMF aligns the ONU to the vOLTMF configuration for the ONU. Likewise, when the vOLTMF identifies that the alarm data store is not current, the vOLTMF requests the current alarm information from the ONU.

5.6.4.1 ONU Datastore Tags

The vOLTMF and vOMCI function maintains the configuration datastore of an ONU. The vOLTMF is the authoritative datastore of the ONU that is represented by the associated ONU YANG modules. The vOMCI function maintains the ONU's representation using ITU-T G.988 OMCI MEs. In order keep the vOLTMF's datastore of an ONU synchronized with the vOMCI's representation of the ONU's datastore, the vOLTMF can optionally use an ONU datastore tag (DST) that represents the latest synchronization state of the ONU's configuration datastore as seen by the vOLTMF. While the method of how the value of the DST is determined is different from how the value of the MIB data sync (MDS) is determined; the usage of ONU datastore tags is like the MDS used by the vOMCI function to synchronize the ONU's representation in the ONU with the representation maintained by the vOMCI function.

When the vOLTMF uses the DST attribute of an ONU's datastore whenever the vOLTMF acts on the ONU's configuration datastore, the vOLTMF uses the `datastore_tag` field in the `ReplaceConfig`, `UpdateConfig`, `RPC` and `Action` messages defined in section 5.7.1. The vOLTMF is able to retrieve the last DST maintained by the vOMCI function using the `datastore_tag` field in the `GetDataResp` message. If the DST is not supported by either the vOMCI function or vOLTMF or has not been set by the vOLTMF, the `datastore_tag` field is defined by the empty string.

5.6.4.2 Requirements

R-vOLTMF.700 - The vOLTMF MUST maintain the configuration data for each ONU.

R-vOLTMF.710 - The vOLTMF MUST align (synchronize) the ONU configuration whenever the vOLTMF detects the ONU is mis-aligned with the configuration maintained by the vOLTMF. One special case includes whenever the vOMCI function has never completed the MIB synchronization function with the ONU.

R-vOLTMF.720 - The vOLTMF MUST align (synchronize) the ONU alarm state whenever the vOLTMF detects the ONU alarm status is mis-aligned.

R-vOLTMF.731 - The vOLTMF MUST provide the capability to align ONU alarms detected by the OLT.

R-vOLTMF.750 – When using Datastore Tags to align the configuration data of an ONU and the vOLTMF acts on the ONU's configuration datastore, the vOLTMF MUST provide a unique, in the context of the ONU, value for the *datastore_tag* field in the *ReplaceConfig*, *UpdateConfig*, *RPC* and *Action* messages defined in section 5.7.1.

5.7 vOLT Management Function to vOMCI Function or vOMCI Proxy Interface (M_{vOLTMF-vOMCI})

The M_{vOLTMF-vOMCI} interface receives YANG configuration data, actions and requests for operational data from the vOLTMF for a target ONU, the vOMCI function or vOMCI Proxy instances. Similarly, the interface is used to transmit the resulting responses from the vOMCI function or vOMCI Proxy back to the vOLTMF. Additionally, the M_{vOLTMF-vOMCI} interface is used by the vOMCI function to send autonomous events or notifications that are emitted from ONUs to the vOLTMF as well as autonomous events and notifications that are for the vOMCI function or vOMCI Proxy entities themselves. Furthermore, the interface is used by the vOLTMF to convey ONU device lifecycle, ONU management chain and topology information to both the vOMCI function and vOMCI Proxy.

The messages conveyed on the M_{vOLTMF-vOMCI} interface are serialized as Google Protobufs (GPB) [38]. The YANG models contained within the messages use TR-385 Issue 2 for ONUs and Annex A: vOMCI YANG Modules for vOMCI function and vOMCI Proxy instances. The YANG models are encoded in JSON using RFC 7951 [17] encoding.

Note: Encoding of YANG models that are directly converted to GPB instead of encoding the YANG models using RFC 7951 is for future study.

5.7.1 Information Model and Messages

The information conveyed on the M_{vOLTMF-vOMCI} interface includes data model messages that are targeted to the ONUs managed by the vOMCI function. Additionally, data model messages can be directed toward the vOMCI function or vOMCI Proxy instances themselves (e.g., ONU device lifecycle, ONU topology information).

The messages used on the interface are comprised of a header and a body. The information in the header is common to all messages and each message within the body has its own set of attributes that comprise the request, response or notification.

```
message Msg {  
    Header header = 1;  
    Body body = 2;  
}
```

Figure 5.7-1/Figure 7 vOLTMF-vOMCI Message

5.7.1.1 Message Header

The message header is comprised of 2 fields which are optional in the protocol.

The *msg_id* field is an optional field used by the vOLTMF to correlate requests with responses. Likewise, the vOMCI function, ONU Management Proxy or vOMCI Proxy can use the *msg_id* field to identify notifications sent by these entities.

The *sender_name* field is an optional field that provides the unique name of the entity that originated the request, response or notification (i.e., vOMCI Proxy, vOMCI function, ONU Management Proxy, vOLTMF).

The *recipient_name* field is an optional field that provides the unique name of the entity that is to receive the request, response or notification (i.e., vOMCI Proxy, vOMCI function, ONU Management Proxy, vOLTMF). The *recipient_name* field is used in certain message transfer protocols like Kafka to verify the intended recipient of the message.

The *object_type* field is a mandatory field that provides the type of object or resource (i.e., vOMCI Proxy, vOMCI function, ONU Management Proxy, ONU) that is the subject of the message.

The *object_name* field is a mandatory field that provides the name of object or resource that is the subject of the message.

Note: The configuration of the names for the vOLTMF, ONU Management Proxy, vOMCI function or vOMCI Proxy instances occurs outside of M_{vOLTMF-vOMCI} interface, such as by command-line parameter or environment variable passed to the vOMCI function or vOMCI Proxy instances at startup.

The *msg_id*, *sender_name* and *recipient_name* fields are useful for logging and troubleshooting purposes as they can help identify interactions between the vOLTMF and the vOMCI function, ONU Management Proxy or vOMCI Proxy.


```

message Header {
    string msg_id = 1;           // Message identifier to
                                // 1) Identify requests and notifications
                                // 2) Correlate requests and response

    string sender_name = 2;      // Unique name of the entity that
                                // originated the message

    string recipient_name = 3;   // The name of the entity that is to
                                // receive the request

    OBJECT_TYPE object_type = 4; // The type of the object or
                                // resource that is subject of the message

    string object_name = 5;      // The name of the object or resource

    enum OBJECT_TYPE {
        ONU = 0;
        VOMCI_FUNCTION = 1;
        VOMCI_PROXY = 2;
        VOLTMF = 3;
        ONU_MANAGEMENT_PROXY = 4;
        VOMCI_FUNCTION_TYPE = 5; //Category of vOMCI function instances
    }
}

```

Figure 5.7-2/Figure 8 vOLTMF-vOMCI HeaderMessage

5.7.1.2 Message Body

The body of the vOLTMF-vOMCI message can be a *request*, *response* or *notification* message. *Request* messages originate from the vOLTMF while *response* and *notification* messages originate from the vOMCI function, ONU Management Proxy or vOMCI Proxy instances.

```

message Body {
    oneof msg_body {
        Request request = 1;
        Response response = 2;
        Notification notification = 3;
    }
}

```

Figure 5.7-3/Figure 9 vOLTMF-vOMCI Message Body

5.7.1.2.1 Request and Response Messages

Request messages originated from the vOLTMF are directed to either the vOMCI function, ONU Management Proxy or vOMCI Proxy instance. Additionally, the vOLTMF can direct the message to a specific ONU when sending a request to the vOMCI function instance.

Response messages are originated from the vOMCI function, ONU Management Proxy or vOMCI Proxy are directed to the vOLTMF that originated the *request* message.

```

message Request {
    oneof req_type {
        Hello hello = 2;
        GetData get_data = 3;
        ReplaceConfig replace_config = 4;
        UpdateConfig update_config = 5;
        RPC rpc = 6;
        Action action = 7;
    }
}

message Response {
    oneof resp_type {
        HelloResp hello_resp = 3;
        GetDataResp get_resp = 4;
        ReplaceConfigResp replace_config_resp = 5;
        UpdateConfigResp update_config_resp = 6;
        RPCResp rpc_resp = 7;
        ActionResp action_resp = 8;
    }
}

```

Figure 5.7-4/Figure 10 vOLTMF-vOMCI Request and Response Messages

5.7.1.2.1.1 Hello Message

This *Hello request* message is originated by the vOLTMF when the vOLTMF establishes a connection to the vOMCI function, ONU Management Proxy or vOMCI Proxy instance. The *service_endpoint_name* field is used to identify the local service endpoint where the vOLTMF can be reached. The *Hello response* message is used by the vOMCI function, ONU Management Proxy or vOMCI Proxy instances to identify the *service_endpoint* that the vOLTMF used to send the *Hello request* message. Additionally, the *Hello response* message

contains capabilities of the server (e.g., support for ONU configuration replicas within the vOMCI function, support for ONU state).

When the vOLTMF sends messages directly to the vOMCI function instances that provide support for replicas of the ONU configuration within the vOMCI function, the vOMCI function would report the *ONU_CONFIG_REPLICA_SUPPORT* capability. If the vOMCI function supports the ability to handle requests that do not use the replica of an ONU, the vOMCI function would report the *ONU_STATE_ONLY_SUPPORT* capability.

If the ONU Management Proxy is deployed, the ONU Management Proxy will report which versions of types of vOMCI functions the ONU Management Proxy supports along with the associated capabilities for that type of vOMCI function. If some types of vOMCI functions provide support for replicas of the ONU configuration within the vOMCI function, the ONU Management Proxy would report the *ONU_CONFIG_REPLICA_SUPPORT* capability for the corresponding type of vOMCI functions. If some other types of vOMCI functions support the ability to handle requests that do not use the replica of an ONU, ONU Management Proxy would report the *ONU_STATE_ONLY_SUPPORT* capability with the type of vOMCI function.

```

message Hello {
    string service_endpoint_name = 1; //The service endpoint the client
                                    // used to establish the session
}

message NFInformation {
    map<string, string> nf_types = 1; //Valid Key types:
                                    // usage-category, vendor-name,
                                    // software-version

    repeated NFCapability capabilities = 2;
    enum NFCapability {
        NO_CAPABILITY_REPORTED = 0;
        ONU_STATE_ONLY_SUPPORT = 1;
        ONU_CONFIG_REPLICA_SUPPORT = 2;
    }
}

message HelloResp {
    string service_endpoint_name = 1; //The service endpoint the server
                                    //used to listen on the session

    repeated NFInformation network_function_info = 2;
                                    // The type information and
                                    // capabilities supported by
                                    // the network function
}

```

Figure 5.7-5/Figure 11 vOLTMF-vOMCI Hello Request and Response Messages

5.7.1.2.1.2 Get Data Message

The *GetData* message is used to retrieve the state information for the entity identified by the *object_name* of the *request* message header. When a vOMCI function provides support for a replica of an ONU configuration, the *GetData* message also returns the associated configuration of the ONU. The *filter* field in the *GetData request* message is used to scope the request to specific elements. Each filter instance contains an expression that represents a NETCONF [10] subtree filter. The *status_response* field in the *GetData response* message is used to convey the status of the message along with any error codes if the request failed for any reason. The usage of the *datastore_tag* field in the *GetData response* message is described in section 5.6.4.1.

```

message GetData {
    repeated bytes filter = 1;
}

message GetDataResp {
    Status status_resp = 1;
    bytes data = 2;
    string datastore_tag = 3;          // Optional: Datastore tag used to
                                      // synchronize the config datastore
}

```

Figure 5.7-6/Figure 12 vOLTMF-vOMCI GetData Request and Response Messages

5.7.1.2.1.3 Replace Configuration Message

The *ReplaceConfig* message is used to completely replace the configuration of the entity identified by the *object_name* of the *request* message header. The *config_inst* field contains the full configuration instance of the targeted entity. The *status_response* field in the *ReplaceConfigResp* response message is used to convey the status of the message along with any error codes if the request failed for any reason. The usage of the *datastore_tag* field in the *ReplaceConfig* message is described in section 5.6.4.1.

Note: vOMCI function, ONU Management Proxy or vOMCI proxy instance implementations ought to limit unnecessary disruptions in the management, control and user plane traffic due to the replacement of a configuration.

```

message ReplaceConfig {
    bytes config_inst = 1;              // Full configuration instance to
                                      // be used as a replacement of what
                                      // exists for the target
    string datastore_tag = 2;          // Optional: Datastore tag used to
                                      // synchronize the config datastore
}

message ReplaceConfigResp {
    Status status_resp = 1;
}

```

Figure 5.7-7/Figure 13 vOLTMF-vOMCI ReplaceConfig Request and Response Messages

5.7.1.2.1.4 Update Configuration Message

The *UpdateConfig* message is used to update the requested part of the configuration of the entity identified by the *object_name* of the *request* message header. When the subject of the request is an ONU there are two ways of updating the configuration depending on the *ONU_CONFIG_REPLICA_SUPPORT* or *ONU_STATE_ONLY_SUPPORT* capabilities of the vOMCI function. The usage of the *datastore_tag* field in the *UpdateConfig* message is described in section 5.6.4.1.

If the vOMCI function supports the *ONU_STATE_ONLY_SUPPORT* capability, the vOLTMF can send the *UpdateConfigInstance* request message. The *current_config_inst* field in this message contains the full configuration of the ONU before the vOLTMF would have applied any requests itself. The *delta_config* field contains a list of changes for nodes within the ONU configuration database. Additionally, each node contains a NETCONF operation to apply to the node.

If the vOMCI function supports the *ONU_CONFIG_REPLICA_SUPPORT* capability, the vOMCI function uses the list of changes for nodes within the replica's ONU configuration contained within the *delta-config* field.

The *status_response* field in the *UpdateConfigResp response* message is used to convey the status of the message along with any error codes if the request failed for any reason.

Note: vOMCI function, ONU Management Proxy or vOMCI proxy instance implementations ought to limit unnecessary disruptions in the management, control and user plane traffic due to the update of a configuration.

```

message UpdateConfigReplica {
    bytes delta_config = 1;           // List of Node changes with the
                                     // associated operation to apply to
                                     // the node
}

message UpdateConfigInstance {
    bytes current_config_inst = 1;    // Full current configuration
                                     // instance
    bytes delta_config = 2;           // List of Node changes with the
                                     // associated operation to apply to
                                     // the node
}

message UpdateConfig {
    oneof req_type {
        UpdateConfigInstance update_config_inst = 1;
        UpdateConfigReplica update_config_replica = 2;
    }
    string datastore_tag = 3;         // Optional: Datastore tag used to
                                     // synchronize the config store
}

message UpdateConfigResp {
    Status status_resp = 1;
}

```

Figure 5.7-8/Figure 14 vOLTMF-vOMCI UpdateConfig Request and Response Messages

5.7.1.2.1.5 RPC and Action Messages

The *RPC* and *Action* messages permits operations to be executed for the entity identified by the *object_name* of the *request* message header. The *input-data* and *output-data* fields contain the instances of YANG elements that are defined for associated YANG RPC and Action. The usage of the *datastore_tag* field in the *RPC* and *Action* messages is described in section 5.6.4.1.

The *status_response* field in the *Action* and *RPC response* messages is used to convey the status of the message along with any error codes if the request failed for any reason.

```
message RPC {
    bytes input_data = 1;
    string datastore_tag = 2;           // Optional: Datastore tag used to
                                        // synchronize the config datastore
}

message RPCResp {
    Status status_resp = 1;
    bytes output_data = 2;
}

message Action {
    bytes input_data = 1;
    string datastore_tag = 2;           // Optional: Datastore tag used to
                                        // synchronize the config datastore
}

message ActionResp {
    Status status_resp = 1;
    bytes output_data = 2;
}
```

Figure 5.7-9/Figure 15 vOLTMF-vOMCI RPC and Action Request and Response Messages

5.7.1.2.1.6 Status and Error Messages

The *Status* message permits vOMCI function or vOMCI Proxy instances to respond to *request* messages. The *status_code* field in the message indicates the request was successful or if there was an error. If there was an error with the request, the Status message contains one or more *Error* messages in the *error* field of the *Status* message. The Error message is based on NETCONF *rpcErrorType* defined in Appendix B of RFC 6241 [10].

```

message Error {           //Type of error as defined in RFC 6241 section 4.3
    string error_type = 1;    // Error type defined in RFC 6241
                               // Appendix B
    string error_tag = 2;    // Error tag defined in RFC 6241
                               // Appendix B
    string error_severity = 3; // Error severity defined in RFC 6241
                               // Appendix B
    string error_app_tag = 4; // Error app tag defined in RFC 6241
                               // Appendix B
    string error_path = 5;   // Error path defined in RFC 6241
                               // Appendix B
    string error_message = 6; // Error message defined in RFC 6241
                               // Appendix B
}

message Status {
    enum StatusCode {
        OK = 0;
        ERROR_GENERAL = 1;
    }
    StatusCode status_code = 1;
    repeated Error error = 2; //Optional: Error information
}

```

Figure 5.7-10/Figure 16 vOLTMF-vOMCI Status Message

The following JSON fragment is an equivalent representation for the first example shown in the Section 4.3 of RFC 6241 [10] and would be encoded in the error field of the *Status* message.

Note: Instead of the "rpc-error" field as define in RFC 6241 [10], the root element is named "error".

Error response

```

"error": {
    "error-type": "rpc",
    "error-tag": "missing-attribute",
    "error-severity": "error",
    "error-info": {
        "bad-attribute": "message-id",
        "bad-element": "rpc"
    }
}

```


5.7.1.2.2 Notification Messages

The *Notification* message permits vOMCI function or vOMCI Proxy instances to send notification events to the vOLTMF. The vOMCI function can send notifications for itself or for an ONU as identified in the *object_name* field of the message header. The vOMCI Proxy instance identifies itself in the *object_name* field of the message header. The YANG notification is contained in the *data* field of the message.

```
message Notification {
    string event_timestamp = 1; // Timestamp in ISO 8601 format
    bytes data = 2;
}
```

Figure 5.7-11/Figure 17 vOLTMF-vOMCI Notification Message

5.7.1.3 ONU Device Lifecycle

The ONU device lifecycle information is used by the vOLTMF to inform the vOMCI function of the creation or deletion of an ONU device, such as by the BAA Device Manager or other Access SDN M&C. An ONU device has to be created in the vOMCI function before it can be configured. Similarly, any configuration for an ONU device in the vOMCI function will be destroyed when the ONU device is deleted.

5.7.1.4 ONU Topology

The ONU topology information is used by the vOLTMF to inform the vOMCI function of the location of the physical ONUs on the Optical Distribution Network (ODN). The ONU topology information includes the ONU device name, the OLT device and Channel Termination to which it is connected, and the assigned TC layer ONU-ID.

The ONU topology information is originated by the physical OLT device as notifications to which the vOLTMF subscribes. The ONU state change notifications include the TC layer ONU serial number as a unique identifier for the ONU along with the ONU state and other topology information. The vOLTMF must correlate the TC layer ONU serial number with a configured ONU device. The vOLTMF then configures the vOMCI function, vOMCI proxy and OLT with the ONU management chain elements that are relevant for that entity. For example, the vOLTMF would configure the ONU's *olt-remote-endpoint* attribute with the remote endpoint that the vOMCI function would use to send vOMCI messages.

5.7.2 Transmission Encoding

The ONU device and the vOMCI function instance information models and associated messages on the vOMCI function's $M_{\text{vOLTMF-vOMCI}}$ interface are encoded in Google Protobufs. Where YANG data is present in the messages (such as the examples in Appendix VIII), the YANG data is encoded within JSON using RFC 7951 [17].

5.7.3 Message Transfer Protocol

The message transfer protocol used for the vOMCI function's $M_{\text{vOLTMF-vOMCI}}$ interface varies based how the vOLTMF and vOMCI function are developed and deployed. This Technical Report defines the requirements for the use of two message transfer protocols Kafka and gRPC.

5.7.3.1 Message Transfer Using Kafka

Kafka is a message transfer protocol used to transfer binary streams of data as event streams using a producer (publication) – consumer (subscription) design pattern. The vOLTMF, ONU Management Proxy, vOMCI function and vOMCI Proxy exchange messages defined in section 5.7.1 (i.e., request, response, notification) by the originating entity publishing the message to a topic on the Kafka bus which is then consumed by the receiving entity that has subscribed to the topic. In order to keep the naming of topics flexible, consumers that subscribe to a topic will always determine if the message is supposed to be for them by inspecting the *recipient_name* fields of the *Request*, *Response* and *Notification* messages. If the message is not for the consumer, the consumer silently discards the message.

5.7.3.1.1 Kafka Topics

The determination of the names that a consumer subscribes, and a producer publishes is configured using the environment in which the vOLTMF and vOMCI function, ONU Management Proxy or vOMCI Proxy is deployed. For example, if the vOMCI function, ONU Management Proxy or vOMCI Proxy is deployed as a docker container, the docker configuration file can contain the topics which the vOMCI message will subscribe and produce. Table 5.7-1 is the list of topic category that vOLTMF and vOMCI function, ONU Management Proxy or vOMCI Proxy will subscribe and produce:

Table 5.7-1/Table 1 Kafka Topic Categories

Topic Category	Description	vOLTMF	vOMCI function/ ONU Management Proxy/vOMCI Proxy
VOMCI_NOTIFICATION	Notifications from the vOMCI function for ONUs and the Message function	Subscribe	Produce
VOMCI_REQUEST	Requests sent by the vOLTMF to the vOMCI function	Produce	Subscribe
VOMCI_RESPONSE	Responses to vOMCI requests	Subscribe	Produce

Note: These are categories of topics and topic names are specific to deployments.

5.7.3.1.2 Requirements

R-M_{vOLTMF-vOMCI}-Kafka.10 – When using Kafka to transfer messages, an entity (i.e., vOLTMF, vOMCI function, ONU Management Proxy, vOMCI Proxy) MUST provide the capability act as a Kafka consumer.

R-M_{vOLTMF-vOMCI}-Kafka.20 – When using Kafka to transfer messages, an entity (i.e., vOLTMF, vOMCI function, ONU Management Proxy, vOMCI Proxy) MUST provide the capability act as a Kafka producer.

R-M_{vOLTMF-vOMCI}-Kafka.30 – When using Kafka to transfer messages, an entity (i.e., vOLTMF, vOMCI function, ONU Management Proxy, vOMCI Proxy) acting as a Kafka consumer MUST silently discard messages for which it is not a recipient by inspecting the *recipient_name* field of the received message when the Kafka producer has provided the *recipient_name* field.

R-M_{vOLTMF-vOMCI}-Kafka.40 – When using Kafka to transfer messages, an entity (i.e., vOLTMF, vOMCI function, ONU Management Proxy, vOMCI Proxy) acting as a Kafka consumer MUST silently discard

messages for which it cannot identify the target recipient by inspecting the *recipient_name* field of the received message when the Kafka producer has provided the *recipient_name* field.

R-M_{vOLTMF-vOMCI}-Kafka.50 – When using Kafka to transfer messages, the vOMCI function or vOMCI Proxy acting as a Kafka consumer MUST silently discard messages for which it cannot identify the target of the request by inspecting the *object_type* and *object_name* fields of the received request message.

R-M_{vOLTMF-vOMCI}-Kafka.60 – When using Kafka to transfer messages, the vOLTMF acting as a Kafka consumer MUST silently discard messages for which it cannot identify the source of the response or notification by inspecting the *object_type* and *object_name* field of the received response or notification message.

R-M_{vOLTMF-vOMCI}-Kafka.70 – Implementations MUST ensure the confidentiality, integrity and authenticity of the Kafka endpoints and the Kafka broker when exchanging messages between the vOLTMF and ONU Management Proxy, vOMCI function or vOMCI Proxy.

R-M_{vOLTMF-vOMCI}-Kafka.80 – When using Kafka to transfer request messages defined in section 5.7.1, the vOMCI solution MAY name Kafka topics that are unique to the ONU Management Proxy, vOMCI function or vOMCI Proxy instances.

R-M_{vOLTMF-vOMCI}-Kafka.90 – When using Kafka to transfer response and notification messages defined in section 5.7.1, the vOMCI solution MAY name Kafka topics that are unique to the vOLTMF.

R-M_{vOLTMF-vOMCI}-Kafka.100 – When using Kafka to transfer messages on the M_{vOLTMF-vOMCI} interface the vOLTMF MUST subscribe and produce messages using the Topic categories defined in Table 5.7-1/Table 1 Kafka Topic.

R-M_{vOLTMF-vOMCI}-Kafka.110 – When using Kafka to transfer messages on the M_{vOLTMF-vOMCI} interface the vOMCI function ONU Management Proxy, or vOMCI Proxy MUST subscribe and produce messages using the Topic categories defined in Table 5.7-1/Table 1 Kafka Topic.

R-M_{vOLTMF-vOMCI}-Kafka.120 – When using Kafka to transfer messages on the M_{vOLTMF-vOMCI} interface, the vOMCI solution MUST control access to the topics for which clients consume and produce messages.

5.7.3.2 Message Transfer Using gRPC

The vOLTMF, vOMCI function and vOMCI Proxy entities can exchange messages defined in section 5.7.1 (i.e., request, response, notification) using the Google Remote Procedure Call (gRPC) [39] message transfer protocol.

5.7.3.2.1 gRPC Channel Initiation

gRPC channels between the vOLTMF and vOMCI function, ONU Management Proxy or vOMCI Proxy is initiated by the vOLTMF to the vOMCI function, ONU Management Proxy or vOMCI Proxy. When the vOLTMF initiates the gRPC channel, the vOLTMF supports the gRPC client interface while vOMCI function, ONU Management Proxy or vOMCI Proxy that support the reception of the messages across the gRPC channel supports the gRPC server interface.

Even though the vOLTMF is the client entity of the gRPC channel, the vOMCI function, ONU Management Proxy or vOMCI Proxy can transmit messages such as Notifications.

5.7.3.2.1.1 gRPC Services

The communication of messages between the vOLTMF and vOMCI function, ONU Management Proxy or vOMCI Proxy are implemented using gRPC Services that permit the vOLTMF to send request messages and receive responses defined in section 5.7.1 using the *VomciMessageNbi* service's *Hello*, *GetData*, *ReplaceConfig*, *UpdateConfig(Replicate|Instance)*, *RPC* or *Action* RPCs. Likewise, vOMCI function, ONU Management Proxy or vOMCI Proxy can send notifications messages defined in section 5.7.1 using the *VomciMessageNbi* service's *ListenforNotification* RPC. The *VomciMessageNbi* service is defined below:

```
service VomciMessageNbi {
    rpc Hello(tr451_vomci_nbi_message.v1.Msg) returns
(tr451_vomci_nbi_message.v1.Msg);

    rpc GetData(tr451_vomci_nbi_message.v1.Msg) returns
(tr451_vomci_nbi_message.v1.Msg);

    rpc ReplaceConfig(tr451_vomci_nbi_message.v1.Msg) returns
(tr451_vomci_nbi_message.v1.Msg);

    rpc UpdateConfigReplica(tr451_vomci_nbi_message.v1.Msg) returns
(tr451_vomci_nbi_message.v1.Msg);

    rpc UpdateConfigInstance(tr451_vomci_nbi_message.v1.Msg) returns
(tr451_vomci_nbi_message.v1.Msg);

    rpc RPC(tr451_vomci_nbi_message.v1.Msg) returns
(tr451_vomci_nbi_message.v1.Msg);

    rpc Action(tr451_vomci_nbi_message.v1.Msg) returns
(tr451_vomci_nbi_message.v1.Msg);

    rpc ListenforNotification (google.protobuf.Empty) returns (stream
tr451_vomci_nbi_message.v1.Msg);
}
```

Figure 5.7-12/Figure 18 vOLTMF-vOMCI VomciMessageNbi Service

5.7.3.2.1.2 Remote Entity Contact Information

When the vOLTMF requests to establish a gRPC channel with the peer entity (i.e., ONU Management Proxy, vOMCI function, vOMCI Proxy), the initiating entity has to have the information necessary to contact the peer entity. This information includes:

- URL of the peer entity (e.g., host name, port)
- Security credentials necessary to establish the identity of the initiating entity

- Security credentials necessary to verify the identity of the peer entity

5.7.3.2.2 gRPC Channel Maintenance

gRPC channels between the vOLTMF and vOMCI function or vOMCI Proxy are considered persistent connections where if the initiation attempt fails or connectivity of the established gRPC channel fails, the gRPC client attempts to reconnect using a reconnection strategy. Likewise, to ensure that a connection is healthy the gRPC client periodically initiates ping requests to ensure the health of the gRPC channel. Likewise, the interval for the HTTP/2 Ping frames that the sent from the gRPC client to server is configurable.

5.7.3.2.3 Using HTTP/2 as the gRPC Wire Protocol

The most common wire protocol used by gRPC is HTTP/2 where many of the attributes of the persistent connection uses underlying features provided by HTTP/2 as defined in RFC 7540 [14] (e.g., keep-alive). The gRPC to HTTP/2 bindings include the values for the request and response headers along with trailers for the HTTP/2 response and are defined by the gRPC specification.

For gRPC requests implementations are able to define and adjust the request timeouts (i.e., `grpc-timeout`) and if compression is used (`grpc-encoding`).

When using HTTP/2 as the gRPC wire protocol, HTTP/2 servers can send a GOAWAY frame to the HTTP/2 client indicating that HTTP/2 server will no longer accept connections. When this occurs the entity, acting as the gRPC and HTTP/2 client will not attempt to re-connect to the peer entity and will provide a notification or log entry that the entity cannot establish communications to the peer entity.

5.7.3.2.4 Securing the gRPC Channel

The gRPC channel between the peer entities is secured using TLS version 1.2 [7] or higher as required by the gRPC specification. This Technical Report uses the TLS session to provide confidentiality and integrity protection of the gRPC channel along with authentication of the entities. The determination of the authenticity of the vOLTMF and vOMCI function, ONU Management Proxy or vOMCI Proxy is provided through the exchange and validation of X.509 certificates.

5.7.3.2.5 Requirements

R-M_{vOLTMF-vOMCI}-gRPC.10 – When using gRPC to transfer messages, the vOLTMF MUST provide the capability to initiate the gRPC channel establishment procedure acting as a gRPC client.

R-M_{vOLTMF-vOMCI}-gRPC.20 - When using gRPC to transfer messages, the vOMCI function MUST provide gRPC server capabilities.

R-M_{vOLTMF-vOMCI}-gRPC.30 - When using gRPC to transfer messages, the vOMCI Proxy MUST provide gRPC server capabilities.

R-M_{vOLTMF-vOMCI}-gRPC.35 - When using gRPC to transfer messages, the ONU Management Proxy MUST provide gRPC server capabilities.

R-M_{vOLTMF-vOMCI}-gRPC.40 - When establishing a gRPC channel between the vOLTMF and vOMCI function, the vOLTMF MUST send the Hello message in order to exchange attachment points and other information needed to exchange messages.

R-M_{vOLTMF-vOMCI}-gRPC.50 - When establishing a gRPC channel between the vOLTMF and vOMCI function, ONU Management Proxy or vOMCI Proxy, the vOLTMF MUST send the ListenForRx RPC in order to allow the peer entity to send messages.

R-M_{vOLTMF-vOMCI}-gRPC.60 - When the vOLTMF acts as a gRPC client, the vOLTMF MUST provide the capability to establish gRPC channels to one (1) or more remote entities using the peer entity's contact information defined in section 5.8.4.1.1 of this Technical Report.

R-M_{vOLTMF-vOMCI}-gRPC.70 - The gRPC channel established between the vOLTMF and vOMCI function, ONU Management Proxy or vOMCI Proxy MUST be a persistent connection where if the initiation attempt fails or the connectivity of the established gRPC fails, the gRPC client attempts to reconnect using the reconnection strategy defined by the gRPC specification.

R-M_{vOLTMF-vOMCI}-gRPC.80 - When the vOLTMF cannot establish a gRPC channel with a remote entity, the initiating entity MUST provide a notification that communication with the remote entity cannot be established along with a possible reason (e.g., HTTP/2 GOAWAY frame).

R-M_{vOLTMF-vOMCI}-gRPC.90 - For established gRPC channels, the vOLTMF MUST ping the remote entity on a periodic interval. The ping interval MUST be configurable and default value MUST be 5 minutes.

R-M_{vOLTMF-vOMCI}-gRPC.100 - When sending gRPC messages, the requesting entity MUST provide a timeout for the request and indicate if compression is used for encoding. The gRPC request timeout and compression values MUST be configurable. The default value for the grpc-timeout MUST be 30 seconds and the default value for the grpc-encoding MUST NOT include compression.

R-M_{vOLTMF-vOMCI}-gRPC.110 – Implementations MUST ensure the confidentiality, integrity and authenticity of the gRPC channels when exchanging messages between the vOLTMF and vOMCI function, ONU Management Proxy or vOMCI Proxy.

R-M_{vOLTMF-vOMCI}-gRPC.120 – The vOLTMF, vOMCI function, ONU Management Proxy and vOMCI Proxy service endpoints MUST implement TLS 1.2 as described in RFC 5246 [7].

R-M_{vOLTMF-vOMCI}-gRPC.130 – When the vOLTMF, acting as the gRPC client, receives the remote entities' X.509 certificate while establishing TLS session, the vOLTMF MUST identify the remote entity according to section 6 of RFC 6125 [9].

R-M_{vOLTMF-vOMCI}-gRPC.140 – The vOMCI function, ONU Management Proxy or vOMCI Proxy, acting as a gRPC server, MUST authenticate the vOLTMF using the X.509 certificate presented by the initiating entity according to section 6 of RFC 6125 [9].

5.7.4 Requirements

R-M_{vOLTMF-vOMCI}.10 - The ONU device data model used on the M_{vOLTMF-vOMCI} interface MUST be compliant with TR-385i2 [35].

R-M_{vOLTMF-vOMCI}.20 - The vOMCI function instance data model used on the M_{vOLTMF-vOMCI} interface MUST be compliant with Annex A: vOMCI YANG Modules.

R-M_{vOLTMF-vOMCI}.30 - The vOMCI Proxy instance data model used on the M_{vOLTMF-vOMCI} interface MUST be compliant with Annex A: vOMCI YANG Modules.

R-M_{vOLTMF-vOMCI}.40 - Messages used on the M_{vOLTMF-vOMCI} interface MUST be encoded using Google Protobufs version 3.0 [38].

R-M_{vOLTMF-vOMCI}.50 – YANG data used with the GPB messages on the M_{vOLTMF-vOMCI} interface MUST be encoded as JSON in RFC 7951 [17].

R-M_{vOLTMF-vOMCI}.60 - Upon reception of the message request, if the vOMCI function, ONU Management Proxy or vOMCI Proxy does not support the version of the message request, the vOMCI function MUST generate an *unsupported-version-vomci-func-message* notification specified in Annex A: vOMCI YANG Modules.

R-M_{vOLTMF-vOMCI}.70 - When a message response is received by the vOLTMF and the vOLTMF does not support the version specified in the message, the vOLTMF MUST log the failed message reception.

R-M_{vOLTMF-vOMCI}.80 - When a message notification is received by the vOLTMF and the vOLTMF does not support the version specified in the message, the vOLTMF MUST log the failed message reception.

R-M_{vOLTMF-vOMCI}.80 - The information model for ONU device data model used on the M_{vOLTMF-vOMCI} interface MUST support the RFC-7895 [15] YANG Modules data module.

5.8 vOMCI Function to OLT Interface (M_{vOMCI-OLT})

The M_{vOMCI-OLT} interface transfers OMCI messages that have been translated by the vOMCI function to the OLT for delivery to the target ONU via the OLT's OMCI channel. Similarly, the M_{vOMCI-OLT} interface is used to receive notifications and operational data from the OLT that originated from the ONU via the OMCI channel.

Furthermore, the M_{vOMCI-OLT} interface delivers requests, responses and operational data that are related to the use of PLOAM service defined in R-OLT-500.

The messages that are conveyed across the M_{vOMCI-OLT} interface are serialized in Google Protocol Buffers (GPB) along with other information elements that help with the forwarding of the message as well as assistance in the operation and troubleshooting of the message flow between the vOMCI function and the OLT. These messages are known as vOMCI messages and are defined in section 5.8.1.

The vOMCI messages that are exchanged between the OLT and vOMCI function can transit through optional, intermediate entities known as vOMCI Proxys.

The vOMCI function, OLT and vOMCI Proxy entities transfer vOMCI messages using the Google RPC (gRPC) message transfer protocol.

5.8.1 vOMCI Messages

The vOMCI message (*VomciMessageSbi*) protobuf definition provides the definition for the request and response messages that are exchanged between the OLT and the vOMCI function. These vOMCI messages can be forwarded through vOMCI Proxy intermediate entities.

The vOMCI message is defined by the GPB *VomciMessage* definition can be an request or a response message defined for:

- OMCI packet transmission
- Error message that is not associated with any vOMCI message response (e.g., message delivery errors)

In parallel, the vOMCI PLOAM service message defined by the GPB *PloamServiceMessage* can be a request or response message pertaining to PLOAM channel service.


```

message VomciMessage {
    string request_payload_id = 1; //Optional: The request payload identifier received
    from the vOLTMF by the vOMCI function.
    oneof msg{
        VomciError error_msg = 2;
        OmciPacket omci_packet_msg = 3;
        OmciPacket tod_packet_msg = 4;
        OltInfoMessage olt_info_msg = 5;
        OmciPacketBulk omci_packet_bulk_msg = 6;
    }
}

message VomciError {

    enum VomciErrorCode {
        ERROR_GENERAL = 0;
        NO_ROUTE_TO_ONU = 1; //The entity does not know which remote endpoint to send the
        message to the ONU
        NO_RESPONSE_FROM_ONU = 2; //The entity did not receive an expected response for a
        request from the ONU
        UNSUPPORTED_REQUEST = 3; //The entity does not support the request
    }

    VomciErrorCode error_code = 1;
    string error_description = 2; //Optional: Error description
    OnuHeader header = 3; //Optional, recommended to be added when the VomciError error_msg
    is a reply on a VomciMessage containing an omci_packet_msg or tod_packet_msg
}

message PloamServiceMessage {
    OnuHeader header = 1;
    oneof msg{
        // G-PON BIP Interfaces
        BipReq bip_counters_req = 2;
        BipRsp bip_counters_rsp = 3;

        // AES Encryption Interfaces
        repeated EncryptPortIdReq encrypt_port_id_req = 4;
        repeated EncryptPortIdRsp encrypt_port_id_res = 5;

        // Alloc-ID Interfaces
        repeated AssignAllocIdReq assign_alloc_id_req = 6;
        repeated AssignAllocIdRsp assign_alloc_id_rsp = 7;

        VomciError error_msg = 8;
    }
}

```

Figure 5.8-1/Figure 19 vOMCI Message Definition

In the Figure 5.8-1 *VomciMessage* message optionally contains the *request_payload_id*, the value of this field can be passed between the vOLTMF and the vOMCI function in order to correlate the requests/responses that are communicated across the $M_{vOLTMF-vOMCI}$ interface with the messages communicated across the $M_{vOMCI-OLT}$ interface.

For vOMCI solutions that correlate *VomciMessage* request and response messages (e.g., vOMCI function, vOMCI Proxy when performing OMCI message retransmission), the vOMCI Proxy and vOMCI function inserts the value of the *request_payload_id* field from the *VomciMessage* response message into the

VomciMessage message prior to transmission of the *VomciMessage* response message. *VomciError* messages may also contain the optional *header* field which must be exactly the same as the *header* field of the corresponding *omci_packet_msg* or *tod_packet_msg* request.

5.8.1.1 ONU Header Message

vOMCI messages that are targeted for an ONU that is attached to an OLT, uses the *OnuHeader* message to convey information related to the ONU.

```
message OnuHeader {  
    string    olt_name = 1; //The OLT name  
    string    chnl_term_name = 2; //The reference identifier to the channel termination of  
the TR-385 vANI  
    uint32    onu_id = 3; //The TC layer ONU-ID identifier of the TR-385 vANI  
}
```

Figure 5.8-2/Figure 20 ONU Header Message

In Figure 5.8-2 the *OnuHeader* message contains the fields (i.e., *olt_name*, *chanl_term_name* and *onu_id*) that are used by the OLT and vOMCI Proxy entities to address an ONU instance.

5.8.1.2 OMCI Packet Message

The *OmciParam* message encapsulates a standard OMCI payload (without the MIC/CRC field) that has been annotated with the corresponding ONU header, message information that assists in entities treatment of the contents of the OMCI Payload. The *OmciParam* messages can be sent in either direction between the OLT, vOMCI Proxy and vOMCI function entities. The *OmciParamRetrans* message is used when transmitting *OmciParam* message requests that originate from the vOMCI function toward the OLT as described in section 5.8.3.

```

message OmcMsgRetrans {
    uint32 t_maxi = 1; //Optional: The maximum threshold value, in milliseconds, for timer
    expiration as defined in ITU-T G.988/B.2. A value of 0 indicates that the field is not
    defined.

    uint32 r_maxi = 2; //Optional: The maximum retries associated with this message as
    defined in ITU-T G.988/B.2 plus 1 (e.g., Value 1 is the first retry) A value of 0
    indicates that the field is not defined.
}

message MessageInfo {
    OmcMsgRetrans omci_msg_retrans = 1; //Optional: OMCI message retransmission applied
}

message OmcIPacket {
    OnuHeader header = 1;
    MessageInfo msg_info = 2; //Optional: Message information
    bytes payload = 3; //OMCI message payload without the CRC/MIC
}

```

Figure 5.8-3/Figure 21 OmcIPacket Message

Figure 5.8-4, Figure 5.8-5 and Figure 5.8-6 provides examples of the message sequence for ONU notifications, ONU commands and a response to an ONU command.

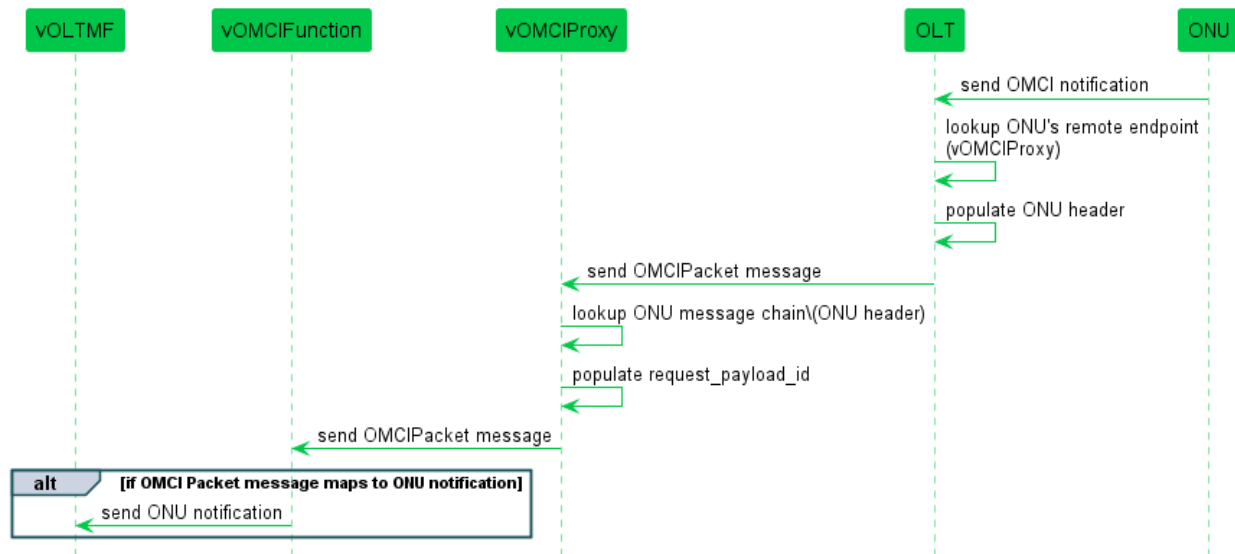
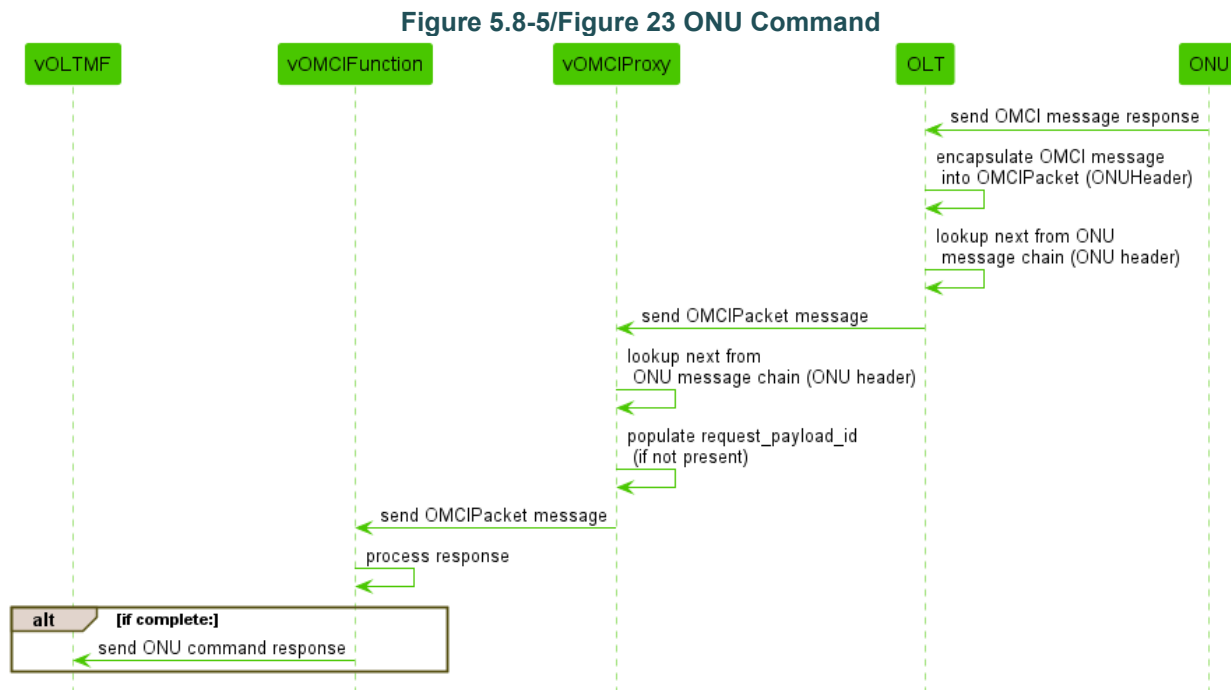
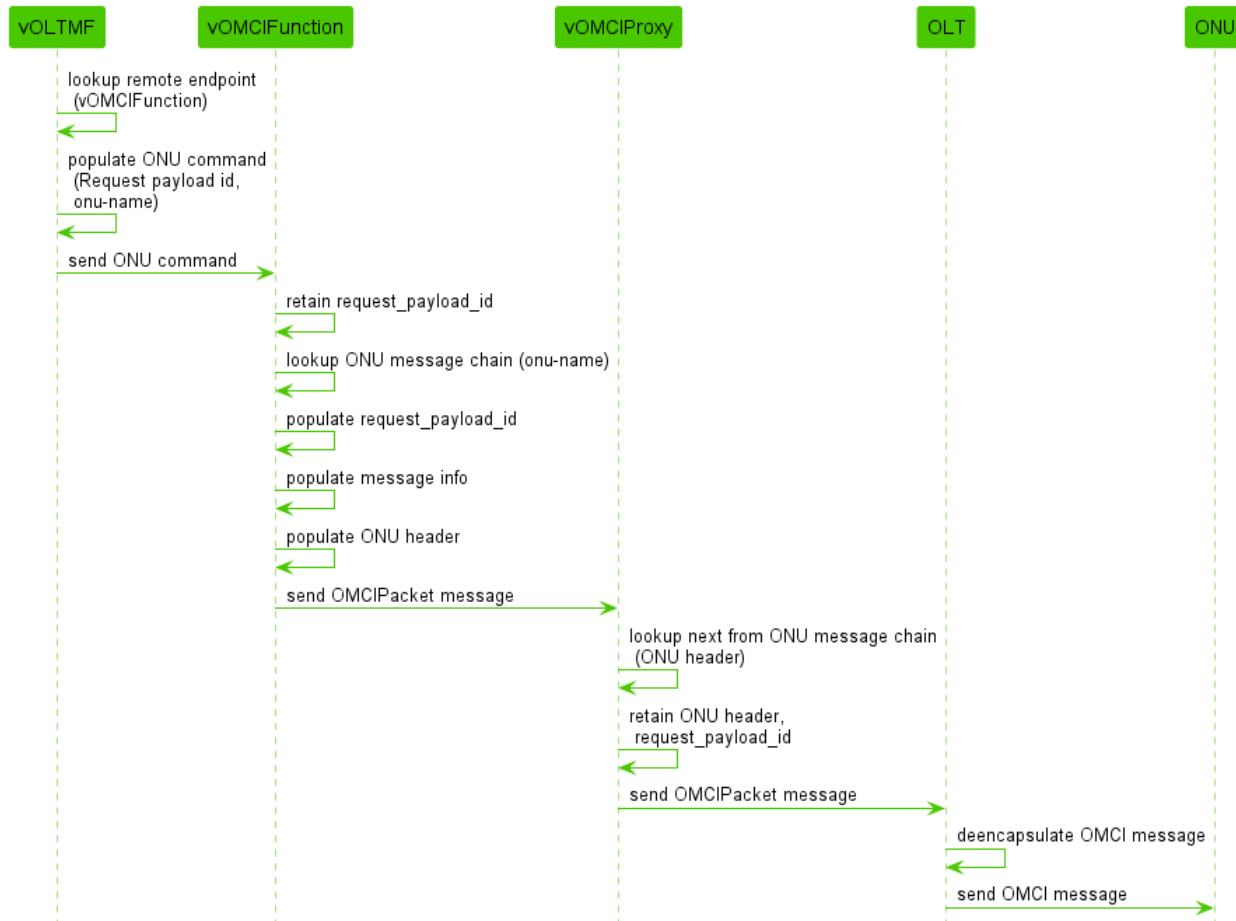


Figure 5.8-4/Figure 22 ONU Notification



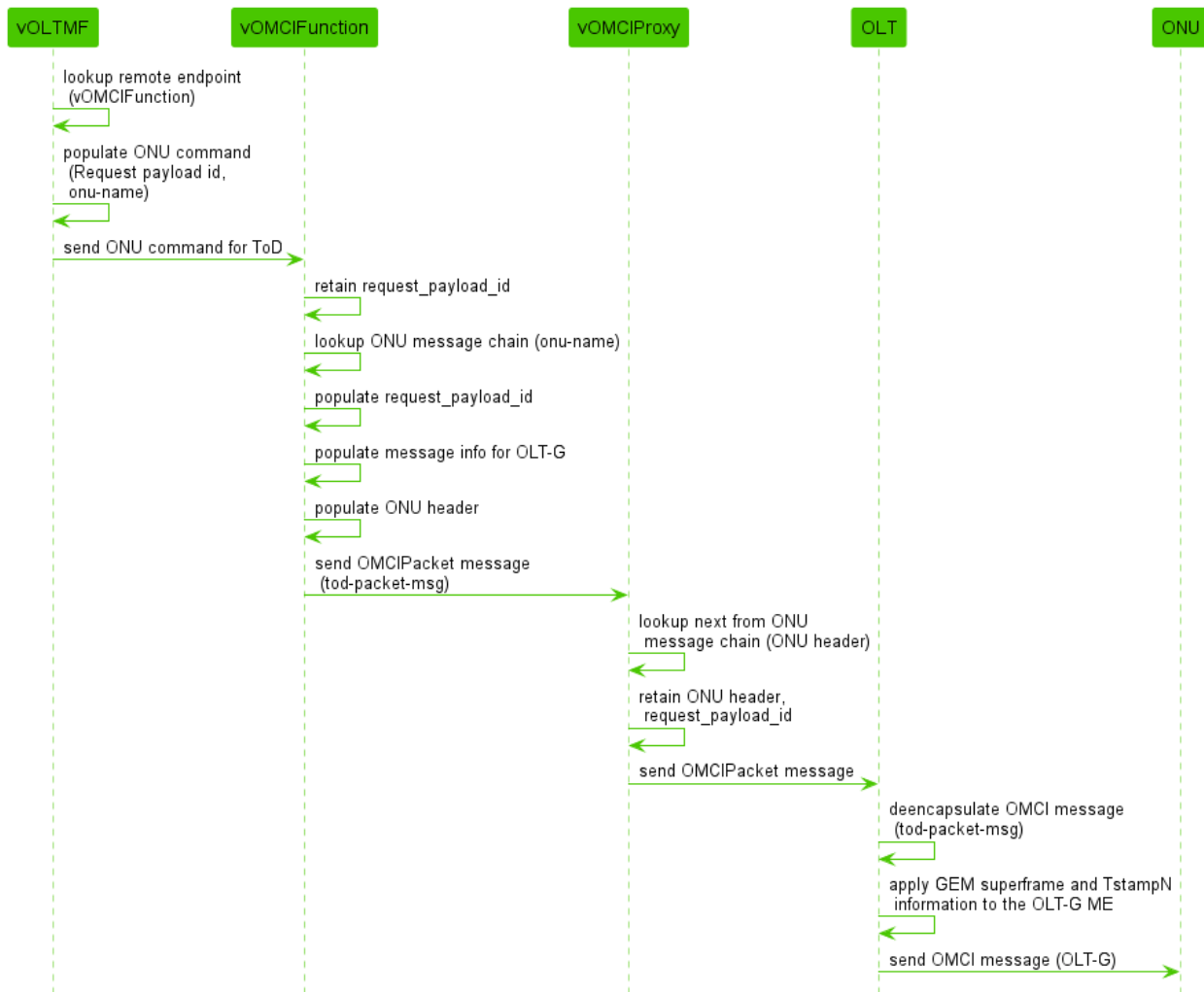


Figure 5.8-7/Figure 25 ToD Packet Message Command

5.8.1.3 vOMCI Message Timeouts

In deployments where the vOMCI Proxy or vOMCI function entities maintains the state of commands that are directed to ONUs, the vOMCI messages that the vOMCI Proxy or vOMCI function tracks could become stale because of various exceptions and timeouts within the deployment. In order to "clean up" the tracking of the command, the vOMCI Proxy or vOMCI function is configured with or utilizes a default value for a timeout that instructs the vOMCI Proxy and vOMCI function to discard any tracking of vOMCI messages that have expired. When the vOMCI Proxy and vOMCI function discards vOMCI messages, the vOMCI Proxy and vOMCI function increments a counter associated with the discard action and log the discard event.

Note: The message timeout described in this section is only applicable when the vOMCI Proxy or vOMCI function entity is not currently performing the OMCI message retransmission function for the vOMCI message. The reason is that the OMCI message retransmission function's timers described in section 5.8.3 take precedence in the "clean up" of the commands over the configured timeout values.

R-vOMCI_Msg_TO.100 – If the vOMCI function or vOMCI Proxy is not currently performing the OMCI message retransmission function but is tracking vOMCI messages from the vOMCI function to an ONU, the vOMCI function or vOMCI Proxy MUST provide the capability to timeout the vOMCI message that it is tracking based on the configured or default timeout value.

R-vOMCI_Msg_TO.110 – If the vOMCI function or vOMCI Proxy times out vOMCI messages from the vOMCI function to an ONU, the vOMCI function or vOMCI Proxy MUST count the timed out message and log the timeout event of the vOMCI message.

5.8.1.4 vOMCI Error Message

In certain error cases, OLT, vOMCI Proxy and vOMCI function entities could have problems in processing an vOMCI message that isn't related to the type of message or the underlying message transfer (e.g., GPB encoding error) but instead is related to the delivery of the vOMCI message to the ONU itself. In this case, the OLT, vOMCI Proxy and vOMCI function can return the vOMCI Error message that describes the problem that was encountered. When present, the optional *header* field must be exactly the same as the *header* field of the request originating the error.

```
message VomciError {
    enum VomciErrorCode {
        ERROR_GENERAL = 0;

        //The entity does not know which remote endpoint to send the message to
        //the ONU
        //For example, to be generated by the OLT in case the chnl_term_name or
        //onu_id is not known in the OLT.
        NO_ROUTE_TO_ONU = 1;

        //The entity did not receive an expected response for a request from the ONU
        //Note that an OLT can be transparent and stateless for OMCI message relay,
        //i.e., it forwards whatever is received in the payload to the ONU without
        //bookkeeping about the messages sent. Such OLT might not know that a
        //response is expected (or not) for a message relayed from the vOMCI to the
        //ONU. As a result such a stateless/transparent OLT relay may not generate
        //this error.
        NO_RESPONSE_FROM_ONU = 2;

        //The entity does not support the request
        //Error could be generated if for example the OMCI message received from the
        //vOMCI has a payload with unexpected size.
        UNSUPPORTED_REQUEST = 3;
    }

    VomciErrorCode error_code = 1;

    //Optional: Error description
    string error_description = 2;

    //Optional, recommended to be added when the VomciError error_msg is a reply
    //on a VomciMessage containing an omci_packet_msg or tod_packet_msg
    OnuHeader header = 3;
}
```

Figure 5.8-8/Figure 26 vOMCI Message Error Message

5.8.1.5 OLT Information Message

When the vOMCI function requires information from the OLT that is needed to assist in communicating information about the OLT to the ONU, the vOMCI function can use the OLT message to retrieve the OLT's information that can then be provided by the vOMCI function to the ONU.

```

message OltInfoMessage {
    oneof msg {
        OltOnuInfoReq olt_onu_info_req = 1;
        OltOnuInfoRsp olt_onu_info_rsp = 2;
    }
}

message OltOnuInfoReq {
    OnuHeader header = 1; //OLT information is scoped toward a specific ONU
}

message OltOnuInfoRsp {
    OnuHeader header = 1; //OLT information is scoped toward a specific ONU
    uint32 gem_superframe_seq = 2; //The sequence number of the specified GEM
    superframe as defined in the OLT-G ME in ITU-T G.988
    string tstampn = 3; //The TstampN as defined in the OLT-G ME in ITU-T G.988
}

```

Figure 5.8-9/Figure 27 OLT Information Message

5.8.1.6 OMCI Packet Bulk Message

The *OmciParamBulk* message is a variation of the *OmciParam*, in which the payload is a repeated field being able to encapsulate a set of OMCI messages. The scope of the message is still a single ONU and hence the payload is annotated with the respective ONU header and potentially message information to facilitate processing or relaying. The message can be optionally used by the vOMCI function, vONU Proxy and OLT entities, whenever their internal logic imposes the need to jointly transmit a set of OMCI messages without waiting for an immediate response between transmissions.

```

message OmcPacketBulk {
    OnuHeader header = 1;

    //Optional: Message information
    MessageInfo msg_info = 2;

    //OMCI message payload without the CRC/MIC
    repeated bytes payload = 3;
}

```

Figure 5.8-10 OMCI Packet Bulk Message

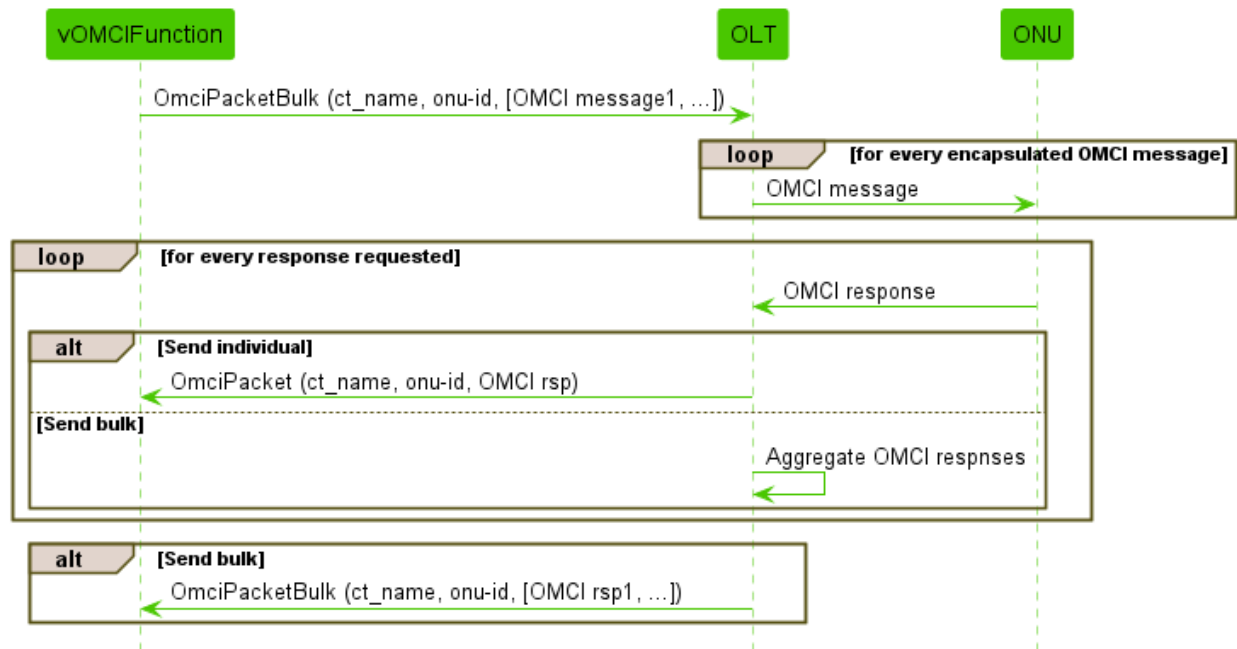


Figure 5.8-11 ONU Command and Response using OMCI Packet Bulk Messages

5.8.1.7 PLOAM Functionality Utilization

The *PloamServiceMessage* message defines a list of functions (delivered over PLOAM) that the vOMCI function can invoke towards the OLT. These are described in the following subsections.

5.8.1.7.1 G-PON BIP Counter

```
message BipReq{
    enum ReqType {
        // Instructs the OLT to enable BER at the ONU and accumulate reports
        // Accumulation period is 15-minutes (similar to other OMCI based PM history
        // counters)
        EnableBipCounter = 0;

        // Instructs the OLT to disable BER at the ONU and stop accumulation
        DisableBipCounter = 1;
        // Requests from the OLT the current accumulated BER value
        RetrieveCurrentCounter = 2;
        // Requests from the OLT the last 15 minutes accumulated BER value
        RetrieveHistoryCounter = 3;
    }
    ReqType bip_specific_req = 1;
}
```

Figure 5.8-12 BIP Request Message

A BIP counter service establishment involves several PLOAM interactions that the vOMCI function could request from the OLT. These are enumerated in the *BipReq* message and one of them should be conveyed in the message (*bip_specific_req*):

- **EnableBipCounter:** The OLT must send the respective PLOAM message to the ONU. The value that corresponds to the BER interval is decided by the OLT following the demands described in section 5.5.5. Simultaneously, the OLT should initialize the local current and history counters.
- **DisableBipCounter:** The OLT terminates the BER interval at the ONU via PLOAM message and stops (even deletes) the related BIP counters maintained for that ONU.
- **RetrieveCurrentCounter / RetrieveHistoryCounter:** Request the current or history values of the BIP counter respectively. The OLT fetches the accumulated values from its memory.

5.8.1.7.2 G-PON BIP Responses

```

message BipRsp {
  oneof msg {
    Status status_rsp = 1;
    // Contains the number of current BIP errors
    uint64 current_bip_counter = 2;
    // Contains the number of BIP errors for the last completed 15 minutes interval
    uint64 history_bip_counter = 3;
  }
}

```

Figure 5.8-13 BIP Response Message

The OLT, depending on the *BipReq* request, can respond with the *BipRsp* message one of the following:

- *status_rsp*: The result of applying the request
- *current_bip_counter*: The so far accumulated value of BIP counter
- *history_bip_counter*: The BIP counter value of the latest complete 15-minute interval

5.8.1.7.3 Alloc-ID Assignment

```

message AssignAllocIdReq {
  enum AllocIdAction {
    // Assign the Alloc-ID to the ONU
    Allocate = 0;
    // Remove the Alloc-ID assigned to the ONU
    Deallocate = 1;
  }
  uint64 alloc_id = 1;
  AllocIdAction allocation_action = 2;
}

message AssignAllocIdRsp {
  uint64 alloc_id = 1;
  Status status_rsp = 2;
}

```

Figure 5.8-14 Allocate/Deallocate Alloc-ID Messages

The vOMCI function can request from the OLT to assign or release an Alloc-ID to/from an ONU via the *AssignAllocIdReq* message.

The OLT shall generate the corresponding PLOAM message towards the ONU and respond back with the result of the function for the respective *alloc_id*, by the *AssignAllocIdRsp* message.

5.8.1.7.4 G-PON AES Encryption

```

message EncryptPortIdReq {
    enum EncryptionAction {
        // Not encrypted or disable encryption
        Disable = 0;
        // Encrypted or enable encryption
        Enable = 1;
    }
    uint64 gem_port_id = 1;
    EncryptionAction encrypt_action = 2;
}
message EncryptPortIdRsp {
    uint64 gem_port_id = 1;
    Status status_rsp = 2;
}

```

Figure 5.8-15 G-PON AES Encryption Messages

Utilizing the *EncryptPortIdReq*, the vOMCI function can request the OLT to generate the proper PLOAM message to enable/disable the encryption of the ONU's GEM port. Once the PLOAM interaction is completed, the OLT shall respond back to the vOMCI function with the result of the function for the specific GEM port on the *EncryptPortIdRsp* message.

5.8.1.7.5 Response Status

```

message Status {
    enum StatusCode {
        OK = 0;
        ERROR_GENERAL = 1;
        // The ID conveyed, belongs to another ONU (intended for Alloc-ID and GEM port ID requests)
        ID_CONFLICT = 2;
    }
    StatusCode status_code = 1;
    string error_description = 2;
}

```

Figure 5.8-16 Response Status Message

The OLT response message for the requests should contain the Status of the action's result, using one of the defined codes.

5.8.2 Encapsulating vOMCI Messages in GPB for use with gRPC Transport

vOMCI messages encapsulated in a GPB message using the schema defined in this section of the Technical Report. GPB message services are defined for communication between the OLT, vOMCI Proxy and vOMCI function entities. These services are identified below:

```
package tr451_vomci_sbi_service.v2;

service VomciHelloSbi {
    rpc HelloVomci (tr451_vomci_sbi_message.v1.HelloVomciRequest) returns
        (tr451_vomci_sbi_message.v1.HelloVomciResponse);
}

service VomciMessageSbi {
    rpc transferOmci (stream tr451_vomci_sbi_message.v1.VomciMessage) returns (stream
        tr451_vomci_sbi_message.v1.VomciMessage);
}

service PloamUtilisationSbi {
    rpc PloamFunction (stream tr451_ploam_sbi.v1.PloamServiceMessage) returns (stream
        tr451_ploam_sbi.v1.PloamServiceMessage);
}
```

Figure 5.8-17/Figure 28 vOMCI SBI Service V2

This Technical Report defines the version 2 of the vOMCI SBI service. In version 2, the RPCs *ListenForVomciRx* and *VomciTx* are replaced by the *transferOmci* RPC for improved performance. The same GPB messages can be used with both versions of the service, so the vOMCI Function and/or vOMCI Proxy can easily interoperate with OLTs that only support version 1.

5.8.2.1 Hello vOMCI Service

The Hello vOMCI service (*VomciHelloSbi*) is used by the OLT, vOMCI Proxy or vOMCI function when contacting the peer entity using the *HelloVomci* RPC. The information that is provided is information that the peer entity needs on subsequent OMCI message exchanges between the OLT, vOMCI Proxy and the vOMCI function. The gRPC client uses the *HelloVomciRequest* message where the OLT, vOMCI function or vOMCI Proxy identifies the endpoint name that the gRPC client advertises when connected to the gRPC server (i.e. gRPC client's local-endpoint). The *HelloVomciResponse* message provided by the gRPC server includes the endpoint name that the server advertises (i.e., gRPC server local-endpoint). Whenever the endpoint name changes, the entities that has been changed must gracefully terminate all sessions between the entities. New sessions will then be re-initiated the sessions using the new endpoint name.

```

message Hello {
    string endpoint_name = 1; //OLT, vOMCI Proxy or vOMCI function endpoint name
    repeated NFCapability capabilities = 2;

    enum NFCapability {
        NO_CAPABILITY_REPORTED = 0;
        BULK_MESSAGE_SUPPORT = 1;
    }
}

message HelloVomciRequest {
    Hello local_endpoint_hello = 1;
}

message HelloVomciResponse {
    Hello remote_endpoint_hello = 1;
}

```

Figure 5.8-18/Figure 29 Hello vOMCI Service V2

5.8.2.2 vOMCI Message Service

The vOMCI message service (*VomciServiceSbi*) provides the RPCs needed to exchange vOMCI messages between the OLT, vOMCI function and vOMCI Proxy that use gRPC to transfer vOMCI messages.

Since the OLT, vOMCI function and vOMCI Proxy entities require a gRPC stream in order to send vOMCI messages, one of the OLT, vOMCI function and vOMCI Proxy entities instantiates a bidirectional stream by invoking the *transferOmci* on the peer entity. The peer entity can then use this stream to send any vOMCI messages as response.

```

service VomciMessageSbi {
    rpc transferOmci (stream tr451_vomci_sbi_message.v1.VomciMessage) returns (stream
tr451_vomci_sbi_message.v1.VomciMessage);
}

```

Figure 5.8-19/Figure 30 vOMCI Message Service V2

5.8.2.2.1 Acknowledgements at OMCI layer vs gRPC

The gRPC layer provides for two-way RPC invocation with input parameters and output result. For encapsulating OMCI messages, the response with result is not required as ITU-T G.988 specification provides its own re-transmission mechanism. Likewise, OMCI messages are correlated using the ITU-T G.988 message identification mechanisms. The entity that originates the OMCI request or notification is responsible for correlating the responses for the OMCI requests or notification.

5.8.2.3 Requirements

R-ENC.10 - An implementation using Google protocol buffers encoding to encode vOMCI messages MUST conform to the schema defined in section 5.8.2.

R-ENC.15 – An implementation that uses gRPC to transfer Google protocol buffers encoded messages MUST conform to the Google Protobuf services defined in section 5.8.2.

R-ENC.20 - When exchanging OmciPacket messages, message re-transmission MUST be implemented according to section B.2 of ITU-T G.988.

R-ENC.30 - When exchanging OmciPacket messages, message correlation MUST be implemented according to section 11.2.1 of ITU-T G.988.

5.8.2.4 vOMCI PLOAM Service

The vOMCI PLOAM service (*PloamUtilisationSbi*) provides the RPCs needed to utilise the PLOAM functions. A bidirectional gRPC stream is defined to ensure a persistent channel between the vOMCI entities for efficient communication with minimum gRPC protocol overhead.

```
service PloamUtilisationSbi {
  rpc PloamFunction (stream tr451_ploam_sbi.v1.PloamServiceMessage) returns (stream tr451_ploam_sbi.v1.PloamServiceMessage);
}
```

Figure 5.8-20 vOMCI PLOAM gRPC Service

An example of how the service can be performed is shown in Figure 5.8-21.

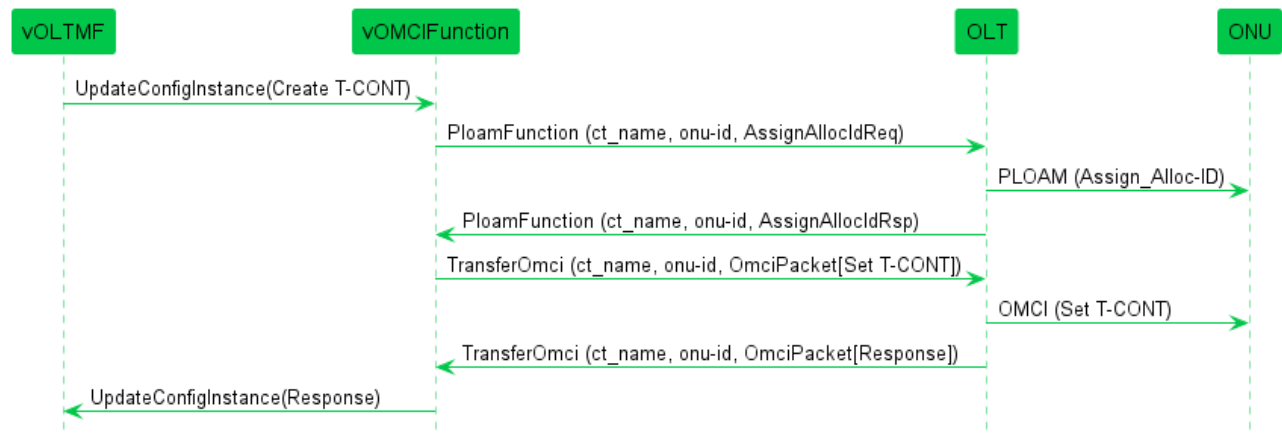


Figure 5.8-21 PLOAM Function Utilization

5.8.3 OMCI Message Retransmission Capability

OmciPacket messages that encapsulate OMCI message requests from the vOMCI function have retransmission that complies with clause B.2.1 of ITU-T G.988 [1]. The OMCI message retransmission capability can be implemented in the vOMCI function or vOMCI Proxy.

In order to implement the retransmission, the vOMCI Proxy and vOMCI function needs to be configured, via its external management interface, with the ITU-T G.988 retransmission (or re-send) thresholds for timer expiration timeout ($T_{\max i}$) and number of retries retry counter ($R_{\max i}$) to be used to control the processing for high priority and low priority messages.

If the vOMCI function performs the OMCI message retransmission and needs to have retransmission behavior that is different from what is configured by the vOLTMF (e.g., MibReset, Software Download), the vOMCI function can optionally use different retransmission elements.

When the retransmission is performed by the vOMCI Proxy and if vOMCI function needs to have special retransmission behavior that is different from what is configured by the vOLTMF (e.g., MibReset, Software Download), the vOMCI function can optionally send the suggested retransmission elements in the *OMCIPacket* message.

Based on configuration from the vOLTMF or using its default configuration, the vOMCI Proxy can accept or ignore the suggested retransmission elements in the *OMCIPacket* message.

Special consideration should be given when OMCI requests are sent aggregated within a single bulk vOMCI message. The timeout threshold at the entity responsible for retransmissions has to take into account that the OMCI messages will be processed individually by the ONU. Thus, the respective responses (as requested by the OMCI messages) will be generated accordingly. It is evident that OMCI request timeout start/end period when sending bulk messages should be relative to the position of the message in the payload.

R-vOMCI_RT.10 – When the vOMCI function initiates the *OMCIPacket* message for an ONU, the vOMCI function SHOULD use the OMCI message retransmission elements (e.g., timer expiration timeout ($T_{\max i}$) and number of retries retry counter ($R_{\max i}$)) for an ONU that the vOMCI function has been configured using default values or those values configured from the vOLTMF.

R-vOMCI_RT.20 – When performing OMCI message retransmission, vOMCI function MAY use different OMCI message retransmission values if the vOMCI function determines other values are needed for successfully completing the OMCI message (e.g., MibReset, Software Download) to an ONU.

R-vOMCI_RT.30 – When the vOMCI Proxy performs OMCI message retransmission, the vOMCI function MAY send OMCI message retransmission values if the vOMCI function determines other values are needed for successfully completing the OMCI message (e.g., MibReset, Software Download) to an ONU.

R-vOMCI_RT.40 – When the vOMCI Proxy is configured to perform OMCI message retransmission for an ONU, the vOMCI Proxy SHOULD use the OMCI message retransmission elements (e.g., timer expiration timeout ($T_{\max i}$) and number of retries retry counter ($R_{\max i}$)) for an ONU that the vOMCI Proxy has been configured using default values or those values configured from the vOLTMF.

R-vOMCI_RT.50 – When the vOMCI Proxy is configured to accept the OMCI retransmission elements in the *OMCIPacket* message, the vOMCI Proxy MUST use the retransmission elements in the *OMCIPacket* message when the retransmission elements are present in the *OMCIPacket* message.

R-vOMCI_RT.60 – When multiple OMCI requests are encapsulated within a bulk vOMCI message, the retransmission timeout values ($T_{\max i}$) MUST be calculated individually for each encapsulated OMCI message. The configured or default retransmission value is applicable to the initial OMCI message. For subsequent OMCI messages within the same bulk vOMCI message, the retransmission value MUST be augmented by an offset which accounts for the previous OMCI message(s)' service time (as an estimation).

5.8.4 Use of gRPC to Exchange GPB Encapsulated vOMCI Messages

OMCI messages that have been encoded using Google Protobufs as described in section 5.8.2 of this Technical Report are transferred between the OLT, vOMCI Proxy function and vOMCI function using Google Remote Procedure Call (gRPC).

5.8.4.1 gRPC Channel Initiation

gRPC channels between the OLT, vOMCI Proxy and vOMCI function can be initiated by any of the OLT, vOMCI Proxy and vOMCI function entities, depending on capabilities (i.e., gRPC client, gRPC server) provided by the OLT, vOMCI Proxy and vOMCI function. OLT, vOMCI Proxy and vOMCI function entities that initiate the gRPC channel supports the gRPC client interface while entities that support the reception of the initiation of the gRPC channel supports the gRPC server interface. OLT, vOMCI Proxy and vOMCI function entities can simultaneously support both the gRPC client and gRPC server interfaces. Regardless of which entity initiates the gRPC channel, since an OLT, vOMCI Proxy and vOMCI function can initiate messages, the asynchronous, bi-directional streaming RPC method is used to exchange messages. When either the OLT, vOMCI Proxy and vOMCI function initiates the gRPC channel, the initiating entity has to send the Hello and Listen RPCs.

Note: In the scenario where both OLT, vOMCI Proxy and vOMCI function entities implements the gRPC client and gRPC server interfaces, the resulting solution will establish two (2) gRPC channel in each direction resulting in two (2) HTTP/2 and associated TCP connections.

5.8.4.1.1 Remote Entity Contact Information

When either the OLT, vOMCI Proxy and vOMCI function requests to establish a gRPC channel with the peer OLT, vOMCI Proxy and vOMCI function, the initiating OLT, vOMCI Proxy and vOMCI function has to have the information necessary to contact the peer entity. This information includes:

- URL of the peer entity (e.g., host name, port)
- Security credentials necessary to establish the identity of the initiating entity
- Security credentials necessary to verify the identity of the peer entity

5.8.4.1.2 OLT, vOMCI Proxy and vOMCI Function Identification and Configuration

Based on the capabilities of OLT, vOMCI Proxy and vOMCI function and depending on the ONUs that are connected to the OLT, OLT, vOMCI Proxy and vOMCI function can establish gRPC channels with multiple instances of peer OLT, vOMCI Proxy and vOMCI function. The determination of peer entity instance(s) that an initiating entity establishes a channel is configured through the provisioning interface of the initiating entity.

5.8.4.2 gRPC Channel Maintenance

gRPC channels between the OLT, vOMCI Proxy and vOMCI function are considered persistent connections where if the initiation attempt fails or connectivity of the established gRPC channel fails, the gRPC client attempts to reconnect using a reconnection strategy. Likewise, to ensure that a connection is healthy the gRPC client periodically initiates ping requests to ensure the health of the gRPC channel. Likewise, the interval for the HTTP/2 Ping frames that the sent from the gRPC client to server is configurable.

5.8.4.3 Using HTTP/2 as the gRPC Wire Protocol

The most common wire protocol used by gRPC is HTTP/2 where many of the attributes of the persistent connection uses underlying features provided by HTTP/2 as defined in RFC 7540 [14] (e.g., keep-alive). The gRPC to HTTP/2 bindings include the values for the request and response headers along with trailers for the HTTP/2 response and are defined by the gRPC specification.

For gRPC requests implementations are able to define and adjust the request timeouts (i.e., `grpc-timeout`) and if compression is used (`grpc-encoding`).

When using HTTP/2 as the gRPC wire protocol, HTTP/2 servers can send a GOAWAY frame to the HTTP/2 client indicating that HTTP/2 server will no longer accept connections. When this occurs the OLT, vOMCI Proxy and vOMCI function, acting as the gRPC and HTTP/2 client will not attempt to re-connect to the peer OLT, vOMCI Proxy and vOMCI function and will provide a notification or log entry that the OLT, vOMCI Proxy and vOMCI function cannot establish communications to the peer OLT, vOMCI Proxy and vOMCI function.

Although the HTTP frames' manipulation is abstracted from the applications through gRPC, HTTP/2 frame size limitations have to be considered when utilizing bulk vOMCI message transmission. The reason is that when the RPC messages exceed the maximum size, the RPC is split to multiple HTTP/2 frames. Thus, it is recommended that the size of the `VomciMessage` does not exceed the maximum HTTP/2 frame size to avoid fragmentation.

5.8.4.4 Securing the gRPC Channel

The gRPC channel between the peer OLT, vOMCI Proxy and vOMCI function entities is secured using TLS version 1.2 [7] or higher as required by the gRPC specification. This Technical Report uses the TLS session to provide confidentiality and integrity protection of the gRPC channel along with authentication of the OLT, vOMCI Proxy and vOMCI function entities. The determination of the authenticity of the OLT, vOMCI Proxy and vOMCI function entities is provided through the exchange and validation of X.509 certificates.

5.8.4.5 Requirements

R-gRPC.10 – When using gRPC to transfer vOMCI messages, a OLT, vOMCI Proxy and vOMCI function SHOULD provide the capability to initiate the gRPC channel establishment procedure acting as a gRPC client.

R-gRPC.20 - When using gRPC to transfer vOMCI messages, a OLT, vOMCI Proxy and vOMCI function SHOULD provide gRPC server capabilities.

R-gRPC.30 - When establishing a gRPC channel between the OLT, vOMCI Proxy and vOMCI function, the Bi-directional streaming RPC method is used to exchange messages.

R-gRPC.40 - When establishing a gRPC channel between the OLT, vOMCI Proxy and vOMCI function, the initiating entity (i.e., OLT, vOMCI function, vOMCI Proxy) MUST send the `HelloVomci` RPC in order to exchange host names and endpoint information needed to exchange messages.

R-gRPC.50 - When establishing a gRPC channel between the OLT, vOMCI Proxy and vOMCI function, the initiating entity (i.e., OLT, vOMCI function, vOMCI Proxy) MUST send the `ListenForVomciRx` RPC in order to allow the peer entity to send vOMCI messages.

R-gRPC.60 - When the OLT, vOMCI Proxy and vOMCI function acts as a gRPC client, the entity MUST provide the capability to establish gRPC channels to one (1) or more remote entities using the peer entity's contact information defined in section 5.8.4.1.1 of this Technical Report.

R-gRPC.70 - The gRPC channel established between the OLT, vOMCI Proxy and vOMCI function MUST be a persistent connection where if the initiation attempt fails or the connectivity of the established gRPC fails, the gRPC client attempts to reconnect using the reconnection strategy defined by the gRPC specification.

R-gRPC.80 - When the OLT, vOMCI Proxy and vOMCI function cannot establish a gRPC channel with a remote entity, the initiating entity MUST provide a notification that communication with the remote entity cannot be established along with a possible reason (e.g., HTTP/2 GOAWAY frame).

R-gRPC.90 - For established gRPC channels, the initiating OLT, vOMCI Proxy and vOMCI function MUST ping the remote entity on a periodic interval. The ping interval MUST be configurable and default value MUST be 5 minutes.

R-gRPC.100 - When sending gRPC requests, the requesting OLT, vOMCI Proxy and vOMCI function MUST provide a timeout for the request and indicate if compression is used for encoding. The gRPC request timeout and compression values MUST be configurable. The default value for the grpc-timeout MUST be 30 seconds and the default value for the grpc-encoding MUST NOT include compression.

R-gRPC.110 – Implementations MUST ensure the confidentiality, integrity and authenticity of the gRPC channels when exchanging messages between the OLT, vOMCI Proxy and vOMCI function.

R-gRPC.120 – The OLT, vOMCI Proxy and vOMCI function endpoints MUST implement TLS 1.2 as described in RFC 5246 [7].

R-gRPC.130 – When an initiating OLT, vOMCI Proxy and vOMCI function, acting as the gRPC client, receives the remote entities' X.509 certificate while establishing TLS session, the initiating entity MUST identify the remote entity according to section 6 of RFC 6125 [9].

R-gRPC.140 – The OLT, vOMCI Proxy and vOMCI function, acting as a gRPC server, MUST authenticate the initiating entity using the X.509 certificate presented by the initiating entity according to section 6 of RFC 6125 [9].

5.9 SMA – OMCI Interface (VOMCI – SMA)

The $M_{\text{VOMCI-SMA}}$ interface conveys the SMA-OMCI messages between the Authenticator and the vOMCI function. The interface messages are serialized in Google Protocol Buffers (GPB) along with other information elements that help with the forwarding of the message.

Optionally, the SMA-OMCI messages can transit through the intermediate vOMCI Proxy entity.

5.9.1 Transport Mechanism

In the most straightforward deployment, SMA – OMCI is performed between the OLT, which operates as the Authenticator and the vOMCI function. Based on 5.8.4, a gRPC channel is established between these entities over which the GPB messages are exchanged.

The same channel can be utilized so that the vOMCI function, OLT and vOMCI Proxy entities transfer SMA-OMCI messages using the Google RPC (gRPC) message transfer protocol.

5.9.2 Messages Definition

The SMA – OMCI messages are defined as a set of GPB messages capable to serve any of the following:

- Initiate the authentication challenge (SMA – OMCI Message 1)

- Respond to the challenge and share the outcome of the process (SMA – OMCI Message 2)
- Confirm the outcome and exchange the result for validation on the other peer (SMA – OMCI Message 3)
- Share the authentication status
- Obtain MSK related information
- Indicate an incorrect message or unexpected behavior with an error message

```

message Msg {
    OnuHeader header = 1;

    oneof msg{
        OltChallenge olt_challenge_req = 2;    // G.988 9.13.11 - Message 1
        OnuAuthRslt onu_authentication_res = 3; // G.988 9.13.11 - Message 2
        OltAuthReq olt_authentication_req = 4;  // G.988 9.13.11 - Message 3
        OnuAuthStatus onu_auth_status_info = 5;
        MSKName msk_name_info = 6;
        VomciError error_msg = 7;
    }
}

```

Figure 5.9-1: SMA – OMCI Messages Definition

At any single time, the originator can transmit one of the GPB messages related to an operation.

5.9.2.1 ONU Header

The *OnuHeader* message contains the appropriate information that uniquely identifies an ONU within the vOMCI entities. The use and definition is the same as in 5.8.1.1.

```

message OnuHeader {
    string    olt_name = 1; //The OLT name
    string    chnl_term_name = 2; //The reference identifier to the channel termination of
the TR-385 vANI
    uint32    onu_id = 3; //The TC layer ONU-ID identifier of the TR-385 vANI
}

```

5.9.2.2 OLT Challenge

The *OltChallenge* message serves the start of the authentication process, by which the Authenticator challenges the ONU and publishes its crypto capabilities.

```

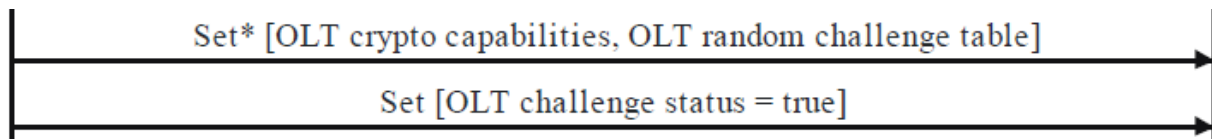
enum CryptoCapabilities {
    NO_CAPABILITY = 0;
    AES_CMACH_128 = 1;
    HMAC_SHA_256 = 2;
    HMAC_SHA_512 = 4;
}

message OltChallenge {
    repeated CryptoCapabilities olt_crypto_cap = 1;
    string olt_random_challenge = 2;
}

```

Figure 5.9-2: OLT Challenge Message

Upon reception of such message, the vOMCI function shall map it to the portion of G.988(10)_F9.13.11-1 [1] OMCI interactions depicted in Figure 5.9-3.

**Figure 5.9-3: OLT Challenge OMCI interactions**

5.9.2.3 ONU Challenge Result

The *OnuAuthRsIt* message returns to the Authenticator the ONU's information on the selected crypto capability and how the crypto key is calculated (so that the OLT can produce the same result).

```

enum CryptoCapabilities {
    NO_CAPABILITY = 0;
    AES_CMACH_128 = 1;
    HMAC_SHA_256 = 2;
    HMAC_SHA_512 = 4;
}

message OnuAuthRsIt {
    CryptoCapabilities onu_crypto_cap = 1;
    string onu_random_challenge = 2;
    string onu_auth_result = 3;
}

```

Figure 5.9-4: ONU Challenge Outcome

In response to the previous message, the ONU will perform internal calculations and store the result into a table. Once the process is completed it will notify the vOMCI function that the data are ready for collection. The vOMCI function, upon reception of the corresponding OMCI AVC, must retrieve from the ONU the appropriate data, form the *OnuAuthRslt* message and transmit it to the Authenticator. The respective G.988(10)_F9.13.11-1 [1] OMCI interactions are summarized in the below Figure 5.9-5.

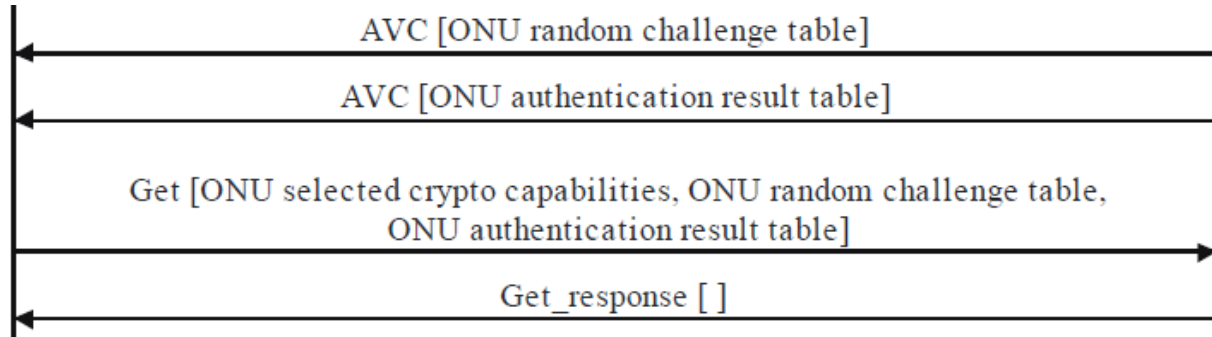


Figure 5.9-5: ONU Response to OLT Challenge

5.9.2.4 OLT Authentication Result

Once the Authenticator collects ONU's response to the challenge, the *OltAuthRslt* message is used from the Authenticator to inform the ONU about the result of the key generation (based on common information).

```
message OltAuthReq {
    string olt_auth_result = 1;
}
```

Figure 5.9-6: OLT Authentication Result

The vOMCI function shall translate the message to the sequence of G.988(10)_F9.13.11-1 [1] OMCI interactions depicted in Figure 5.9-7.

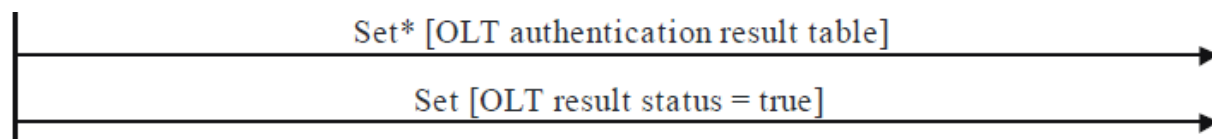


Figure 5.9-7: Set OLT Authentication Result at the ONU

5.9.2.5 ONU Authentication Status

Completing the SMA-OMCI, the authentication status of the ONU must be changed to indicate the communication state. Moreover, the ONU's authentication status may change at any time (for reasons outside of scope of this text). In both cases, the ONU must notify the Authenticator respectively. In addition, the Authenticator might need to spontaneously examine the ONU's authentication status. All these operations are provided through the *OnuAuthStatus* message described below.

```
message OnuAuthStatus {  
    enum ACTION {  
        REQ = 0;  
        REPORT = 1; // The action can be used as a response to an ONU authentication state  
        request or to notify a state change  
    }  
  
    enum AUTH_STATUS {  
        INDETERMINATE = 0;  
        SUCCESS = 3;  
        FAILURE = 4;  
    }  
  
    ACTION auth_state_operation = 1;  
    AUTH_STATUS onu_auth_state = 2; // Optional. Valid only for ACTION code 2  
}
```

Figure 5.9-8: ONU Authentication Status Notification and Retrieval

5.9.2.6 MSK Name

Following the successful SMA-OMCI, the OLT and the ONU have computed the master session key (MSK), which is used for later session encryption derivation. The *MSKName* message can be used by the OLT to retrieve the MSK name from the ONU for further calculations.

```
message MSKName {  
    enum ACTION {  
        REQ = 0;  
        RES = 1;  
    }  
  
    ACTION msk_operation = 1;  
    string msk_info = 2; // Optional. Valid only for ACTION code 2  
}
```

Figure 5.9-9: MSK Name Retrieval

5.9.2.7 Error

During the SMA-OMCI process, there might be situations where one of the involved entities is unable to either interpret or execute the requested message (e.g., due to service unavailability, incapability to collect the appropriate data, etc.). In such cases the *VomciError* shall be flagged so that the other end becomes aware of the situation. The use is the same as defined in 5.8.1.7.

```

message VomciError {
    enum VomciErrorCode {
        ERROR_GENERAL = 0;

        //The entity does not know which remote endpoint to send the message to
        //the ONU
        //For example, to be generated by the OLT in case the chnl_term_name or
        //onu_id is not known in the OLT.
        NO_ROUTE_TO_ONU = 1;

        //The entity did not receive an expected response for a request from the ONU
        //Note that an OLT can be transparent and stateless for OMCI message relay,
        //i.e., it forwards whatever is received in the payload to the ONU without
        //bookkeeping about the messages sent. Such OLT might not know that a
        //response is expected (or not) for a message relayed from the
        //ONU. As a result such a stateless/transparent OLT relay may not generate
        //this error
        NO_RESPONSE_FROM_ONU = 2;

        //Not enough information to form an imminent message
        INCOMPLETE_DATA = 3;

        //The entity does not support the request
        //Error could be generated if for example the OMCI message received from the
        // vOMCI has a payload with unexpected size.
        UNSUPPORTED_REQUEST = 4;
    }

    VomciErrorCode error_code = 1;

    //Optional: Error description
    string error_description = 2;

    //Optional, recommended to be added when the VomciError error_msg is a reply
    //on a VomciMessage containing an omci_packet_msg or tod_packet_msg
    OnuHeader header = 3;
}

```

Figure 5.9-10: SMA-OMCI Error Message

5.9.3 Encapsulate Messages in gRPC Transport

SMA-OMCI GPB messages defined above, can be encapsulated in a dedicated gRPC service exposed by the OLT, vOMCI Proxy and vOMCI function entities. The service is defined as follows:

```
service Sma {  
  
    rpc transferSma (stream tr451_vomci_sma_message.v1.Msg) returns (stream  
        tr451_vomci_sma_message.v1.Msg);  
  
}
```

Figure 5.9-11: SMA-OMCI gRPC Service

The service defines the *transferSma* RPC, which encapsulates SMA-OMCI messages defined in previous subsections.

Annex A: vOMCI YANG Modules

This section describes the YANG modules to be used when managing the vOMCI function, ONU Management Proxy, vOMCI Proxy function, OLT and vOLTMF for the purposes of this Technical Report. These functions use the following YANG modules that are common to these managed entities or are used from other Broadband Forum Technical Reports and IETF RFCs:

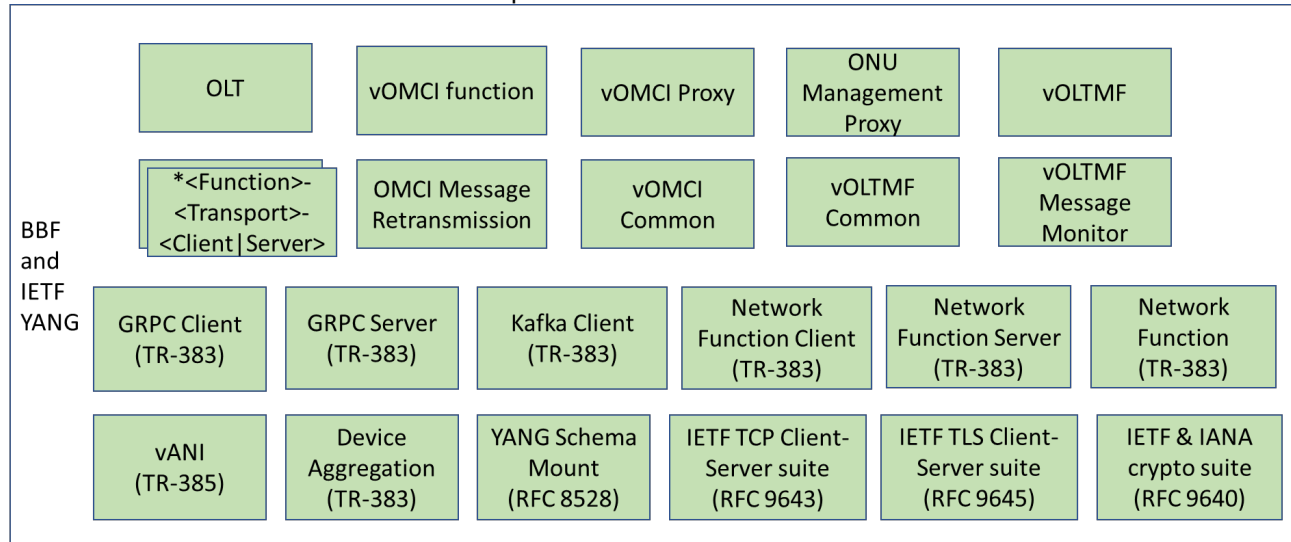


Figure A-1/Figure 31 vOMCI YANG Modules

The <Function>-<Transport>-<Client|Server> YANG modules in Figure A-1 depicts the transport specific YANG modules supported by each of the functions in the vOMCI solution. These modules include:

- bbf-olt-vomci-grpc-client.yang, bbf-olt-vomci-grpc-server.yang
- bbf-onu-management-proxy-grpc-client.yang, bbf-onu-management-proxy-grpc-server.yang, bbf-onu-management-proxy-kafka-agent.yang
- bbf-voltmf-grpc-client.yang, bbf-voltmf-kafka-agent.yang
- bbf-vomci-function-grpc-client.yang, bbf-vomci-function-grpc-server.yang, bbf-vomci-function-kafka-agent.yang
- bbf-vomci-proxy-grpc-client.yang, bbf-vomci-proxy-grpc-server.yang, bbf-vomci-proxy-kafka-agent.yang. Further, for each <Client|Server> file, there are optional augments that provide either TCP-only, or TCP with TLS transport choices.
- Without these optional TCP/TLS augments, by default the only option for each <Client|Server> endpoint is a TCP port number (for example, to be used for localhost or forwarding scenarios outside the scope of YANG).
- To support TCP and/or TLS, the YANG server for any client or server endpoint simply needs to include the correspondingly named YANG files. Exact names are listed in the sub-sections below.

A.1 OLT YANG Modules

The OLT YANG module defines the management data model needed to:

- Establish gRPC endpoints for the vOMCI Proxy and/or vOMCI function.
- Configuration and state information for the ONU management chain that identifies the remote endpoint toward the vOMCI Proxy and/or vOMCI function.

The OLT model is defined in the following modules located in the application directory on Github:

- bbf-olt-vomci.yang
- bbf-olt-vomci-state.yang

- bbf-olt-vomci-grpc-client-tcp.yang
- bbf-olt-vomci-grpc-client-tls.yang
- bbf-olt-vomci-grpc-client.yang
- bbf-olt-vomci-grpc-server-tcp.yang
- bbf-olt-vomci-grpc-server-tls.yang
- bbf-olt-vomci-grpc-server.yang

The OLT model uses the following YANG modules that are common to these managed entities or are used from other Broadband Forum Technical Reports and IETF RFCs:

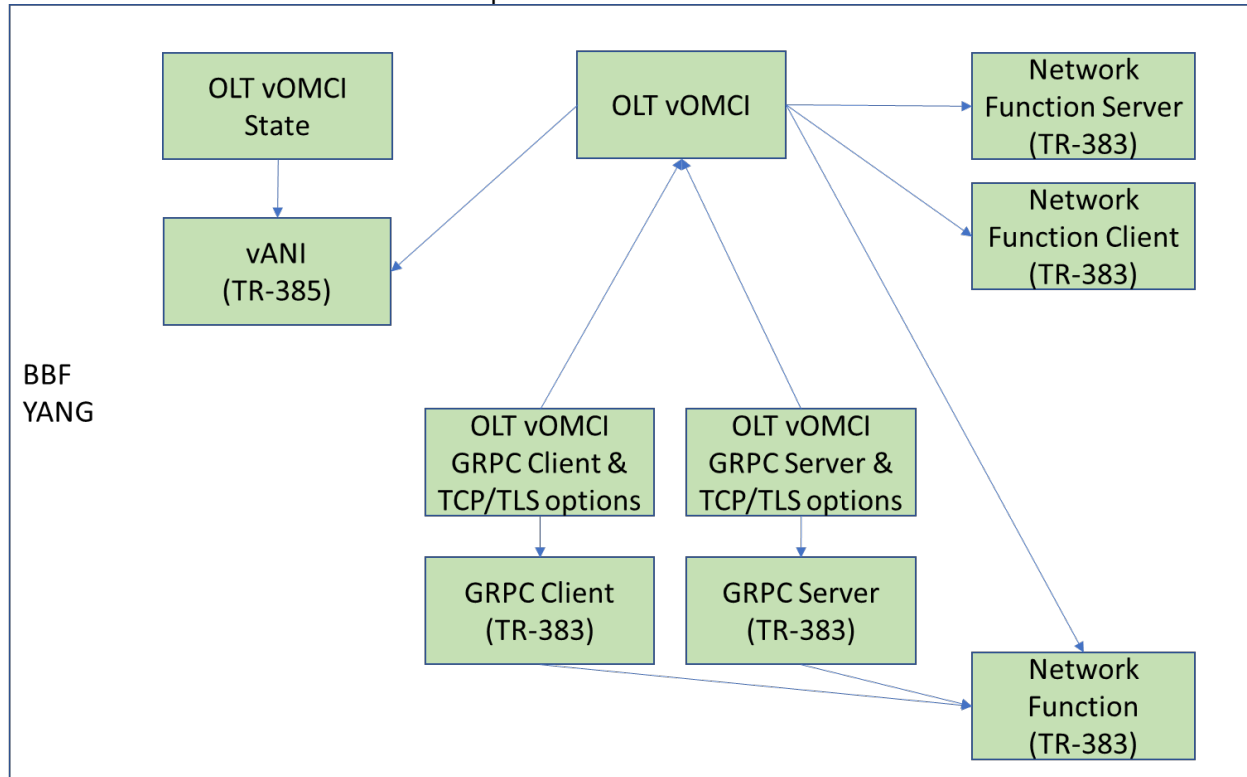


Figure A-2/Figure 32 OLT vOMCI YANG Modules

A.2 vOLTMF YANG Modules

The vOLTMF YANG module defines the management data model needed to:

- Maintain the network function topology for the vOMCI solution.
- Establish gRPC endpoints for the ONU Management Proxy, vOMCI Proxy function, pOLT and vOMCI function.
- Maintain the ONU metadata for the configuration and state information associated with the vOMCI solution. Supports both stand-alone and embedded ONU management models.
- Notification for discovered ONUs with associated device and software information.

The vOMCI function model is defined in the following modules located in the application directory on Github:

- bbf-voltmf.yang
- bbf-voltmf-state.yang
- bbf-voltmf-grpc-client-tcp.yang
- bbf-voltmf-grpc-client-tls.yang
- bbf-voltmf-grpc-client.yang
- bbf-voltmf-kafka-agent-tcp.yang
- bbf-voltmf-kafka-agent-tls.yang

- bbf-voltmf-kafka-agent.yang

The vOLTMF model uses the following YANG modules that are common to these managed entities or are used from other Broadband Forum Technical Reports and IETF RFCs:

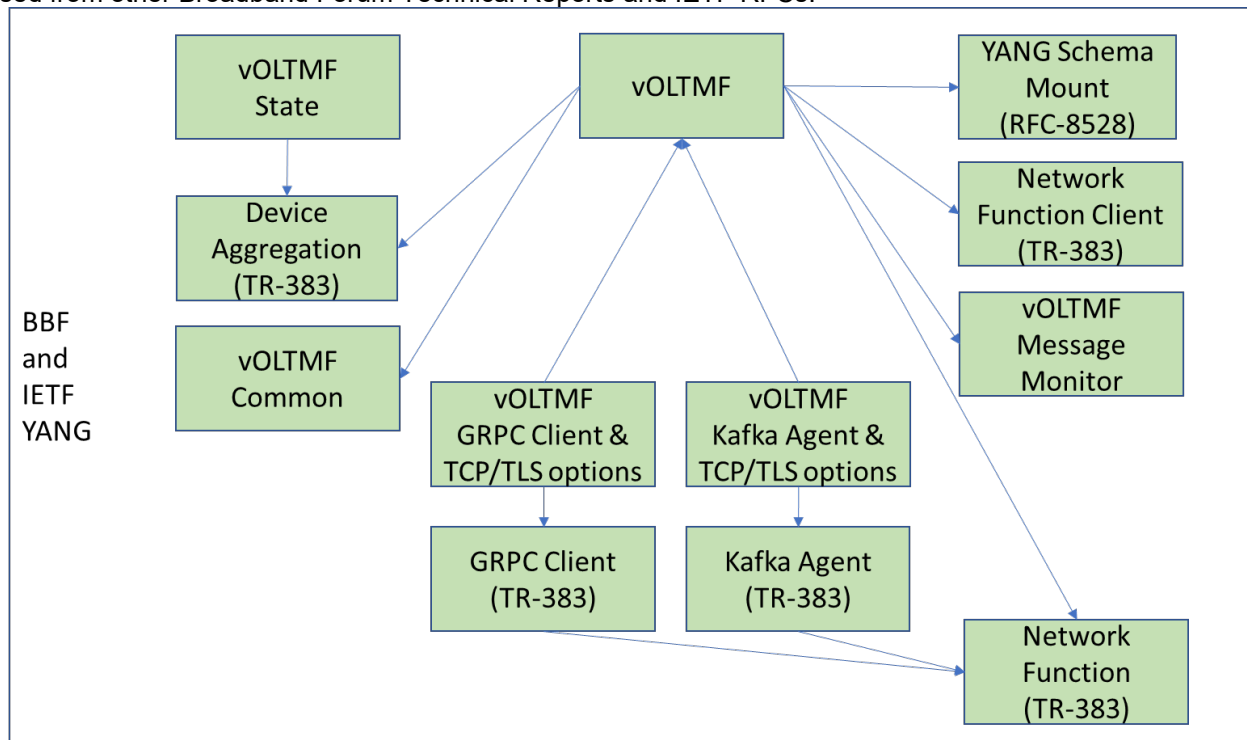


Figure A-3/Figure 33 vOLTMF YANG Modules

A.3 vOMCI function YANG Modules

The vOMCI function YANG module defines the management data model needed to:

- Configuration and state information for ONUs using the vOMCI function.
- Establish gRPC endpoints for the vOMCI Proxy and OLT.
- Configuration and state information for the ONU management chain that identifies the remote endpoint toward the OLT.
- Configuration and state for the OMCI retransmission capability.
- Notifications for:
 - Alignment result of an ONU.
 - ONU misalignment alarm.
 - When the vOMCI function cannot translate a message from the ONU.
 - When the vOMCI function cannot translate a message from the vOLTMF or ONU Management Proxy.

The vOMCI function model is defined in the following modules located in the application directory on Github:

- bbf-vomci-function.yang
- bbf-vomci-function-grpc-client-tcp.yang
- bbf-vomci-function-grpc-client-tls.yang
- bbf-vomci-function-grpc-client.yang
- bbf-vomci-function-grpc-server-tcp.yang
- bbf-vomci-function-grpc-server-tls.yang
- bbf-vomci-function-grpc-server.yang
- bbf-vomci-function-kafka-agent-tcp.yang

- bbf-vomci-function-kafka-agent-tls.yang
- bbf-vomci-function-kafka-agent.yang

The vOMCI function model uses the following YANG modules that are common to these managed entities or are used from other Broadband Forum Technical Reports and IETF RFCs:

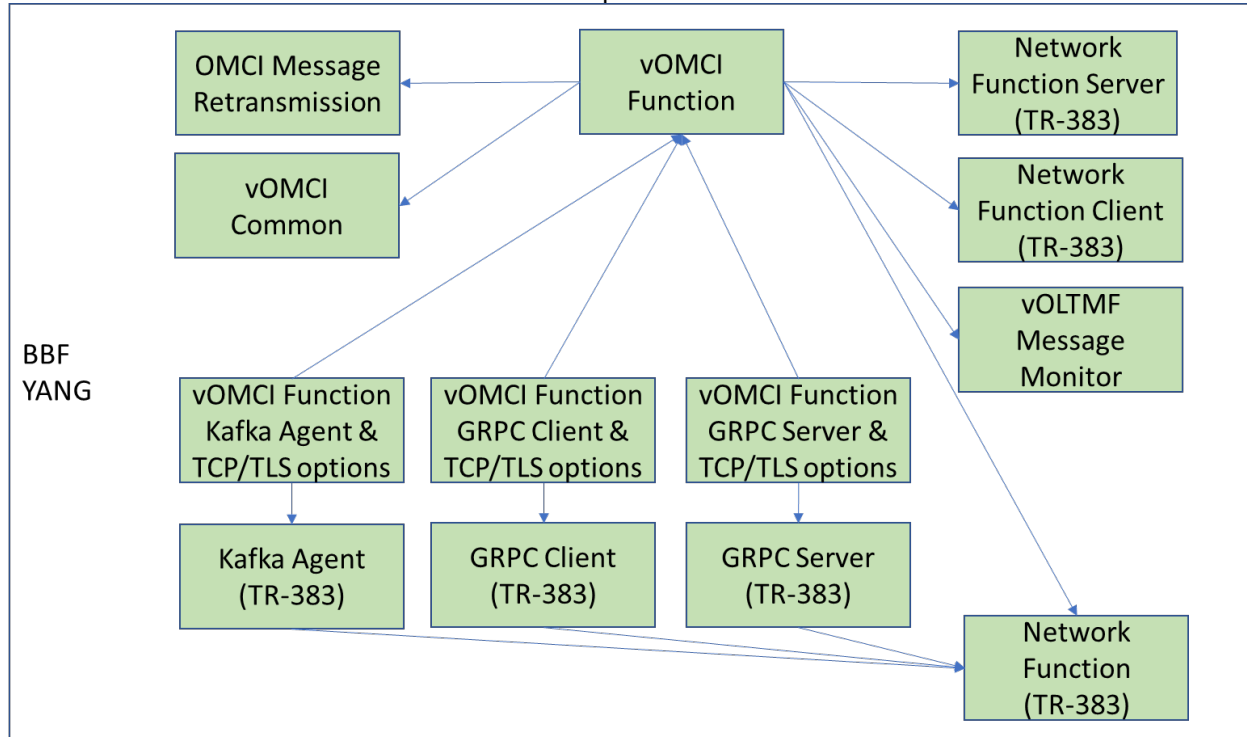


Figure A-4/Figure 34 vOMCI Function YANG Modules

A.4 ONU Management Proxy function YANG Modules

The ONU Management Proxy function YANG module defines the management data model needed to:

- Maintain the network function topology for the vOMCI solution.
- Establish gRPC endpoints for the vOLTMF instances and vOMCI function instances.
- Configuration and state for the association policy between ONUs and vOMCI function instances.
- Configuration and state information for the ONU management chain that identifies the remote endpoints toward the vOLTMF instance and the vOMCI function instance for an ONU.
- State information of the forwarding table used to associate an ONU to a vOLTMF and vOMCI function instance.
- Configuration and state information for ONUs using the ONU Management Proxy.

The ONU Management Proxy model is defined in the following modules located in the application directory on Github:

- bbf-onu-management-proxy.yang
- bbf-onu-management-proxy-state.yang
- bbf-onu-management-proxy-grpc-client-tcp.yang
- bbf-onu-management-proxy-grpc-client-tls.yang
- bbf-onu-management-proxy-grpc-client.yang
- bbf-onu-management-proxy-grpc-server-tcp.yang
- bbf-onu-management-proxy-grpc-server-tls.yang
- bbf-onu-management-proxy-grpc-server.yang
- bbf-onu-management-proxy-kafka-agent-tcp.yang
- bbf-onu-management-proxy-kafka-agent-tls.yang

- bbf-onu-management-proxy-kafka-agent.yang

The ONU Management Proxy model uses the following YANG modules that are common to these managed entities or are used from other Broadband Forum Technical Reports and IETF RFCs:

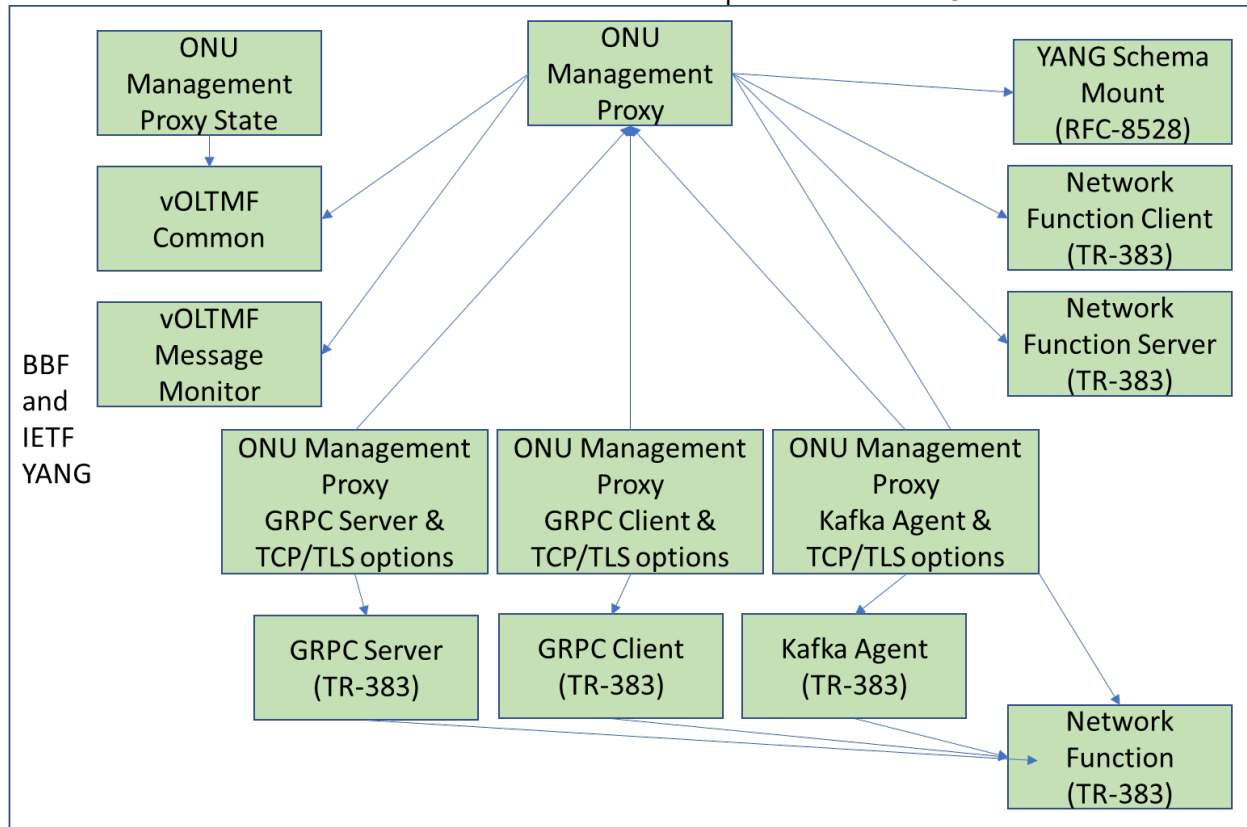


Figure A-5/Figure 35 ONU Management Proxy YANG Modules

A.5 vOMCI Proxy function YANG Modules

The vOMCI Proxy function YANG module defines the management data model needed to:

- Establish gRPC endpoints for the pOLT and vOMCI function.
- Configuration and state for the OMCI retransmission capability.
- Configuration and state information for the ONU management chain that identifies the remote endpoint toward the OLT and the vOMCI function for an ONU.
- Configuration and state information for ONUs using the vOMCI Proxy.

The vOMCI Proxy model is defined in the following modules located in the application directory on Github:

- bbf-vomci-proxy.yang
- bbf-vomci-proxy-grpc-client-tcp.yang
- bbf-vomci-proxy-grpc-client-tls.yang
- bbf-vomci-proxy-grpc-client.yang
- bbf-vomci-proxy-grpc-server-tcp.yang
- bbf-vomci-proxy-grpc-server-tls.yang
- bbf-vomci-proxy-grpc-server.yang
- bbf-vomci-proxy-kafka-agent-tcp.yang
- bbf-vomci-proxy-kafka-agent-tls.yang
- bbf-vomci-proxy-kafka-agent.yang

The vOMCI Proxy model uses the following YANG modules that are common to these managed entities or used from other Broadband Forum Technical Reports and IETF RFCs:

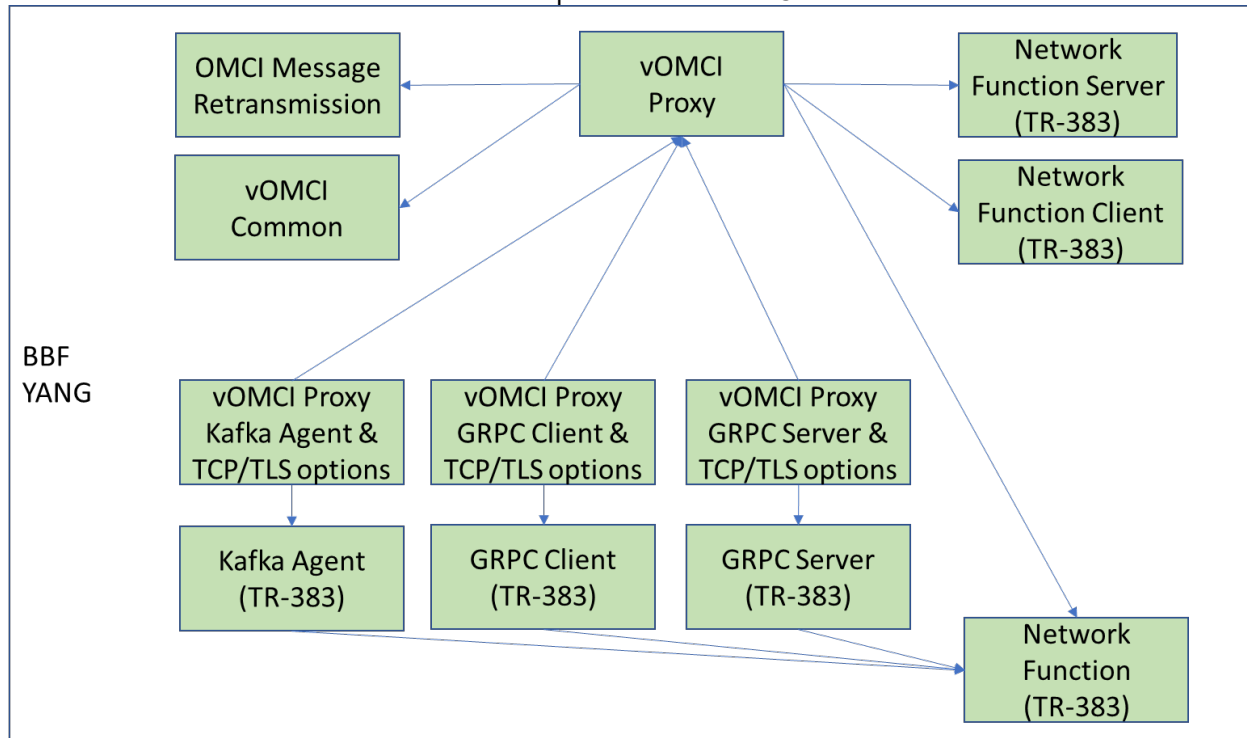


Figure A-6/Figure 36 vOMCI Proxy YANG Modules

A.6 Dependencies on Related YANG Modules and Standards

The YANG modules in this Technical Report are based on YANG 1.1 (RFC 7950 [16])

The following YANG modules are used by this Technical Report:

- Files from TR-383 [33]:
 - bbf-grpc-client.yang
 - bbf-grpc-server.yang
 - bbf-kafka-agent.yang
 - bbf-network-function-client.yang
 - bbf-network-function-server.yang
 - bbf-network-function.yang
 - bbf-device-aggregation.yang
- Files from TR-385 [35]:
 - bbf-xponvni.yang
- Files from upstream sources:
 - ietf-yang-schema-mount.yang [20]
 - iana-crypt-hash.yang [12]
 - iana-tls-cipher-suite-algs.yang [26]
 - ietf-crypto-types.yang [22]
 - ietf-inet-types.yang [11]
 - ietf-keystore.yang [24]
 - ietf-tcp-client.yang [25]
 - ietf-tcp-common.yang [25]
 - ietf-tcp-server.yang [25]
 - ietf-tls-client.yang [26]
 - ietf-tls-common.yang [26]
 - ietf-tls-server.yang [26]

- ietf-truststore.yang [23]
- ietf-x509-cert-to-name.yang [13]
- ietf-yang-types.yang [11]

Appendix I.vOMCI Deployment in CloudCO

The TR-384 [34] CloudCO framework provides the capability to manage Access PNFs using the Access SDN M&C and the BAA layer or directly via the Access SDN M&C's management function as depicted in Figure I-1.

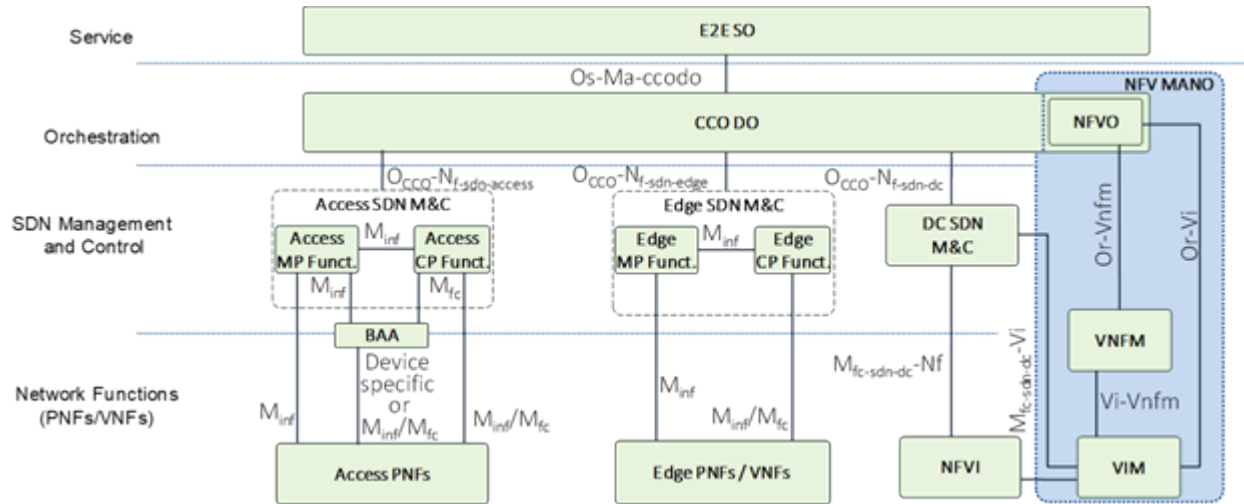


Figure I-1/CloudCO Framework

When deploying the vOMCI solution in the CloudCO framework, the vOLT Management Function is performed by the BAA layer or directly by the Access SDN M&C.

I.1 Integrating the vOMCI Solution with the BAA layer

The CloudCO's BAA layer provides a persistent, digital representation of the Access PNFs regardless of the operational status (e.g., online, offline) of the PNF. In addition, the BAA layer provides the adaptation of the PNFs with device specific interfaces into the standard YANG representations that are used by the Access SDN M&C across the M_{inf} interface. Several functions within the BAA Core along with the vOLTMF provide the capabilities defined by the vOLT Management Function specified in section 5.6 of this Technical Report. Likewise, the vOMCI function, ONU Management Proxy and vOMCI Proxy are implemented as micro-services where they communicate to the vOLTMF via the vOLTMF adapter that complies with the M_{vOLTMF-vOMCI} interface. These functions and interfaces are depicted in Figure I-2.

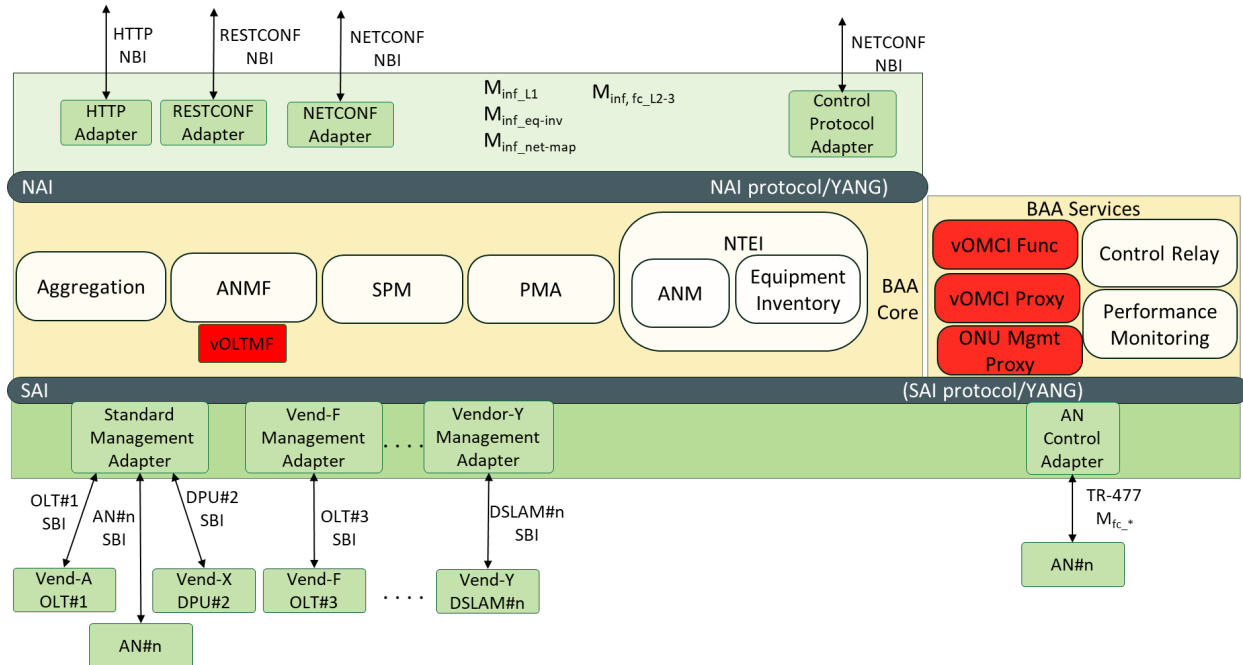
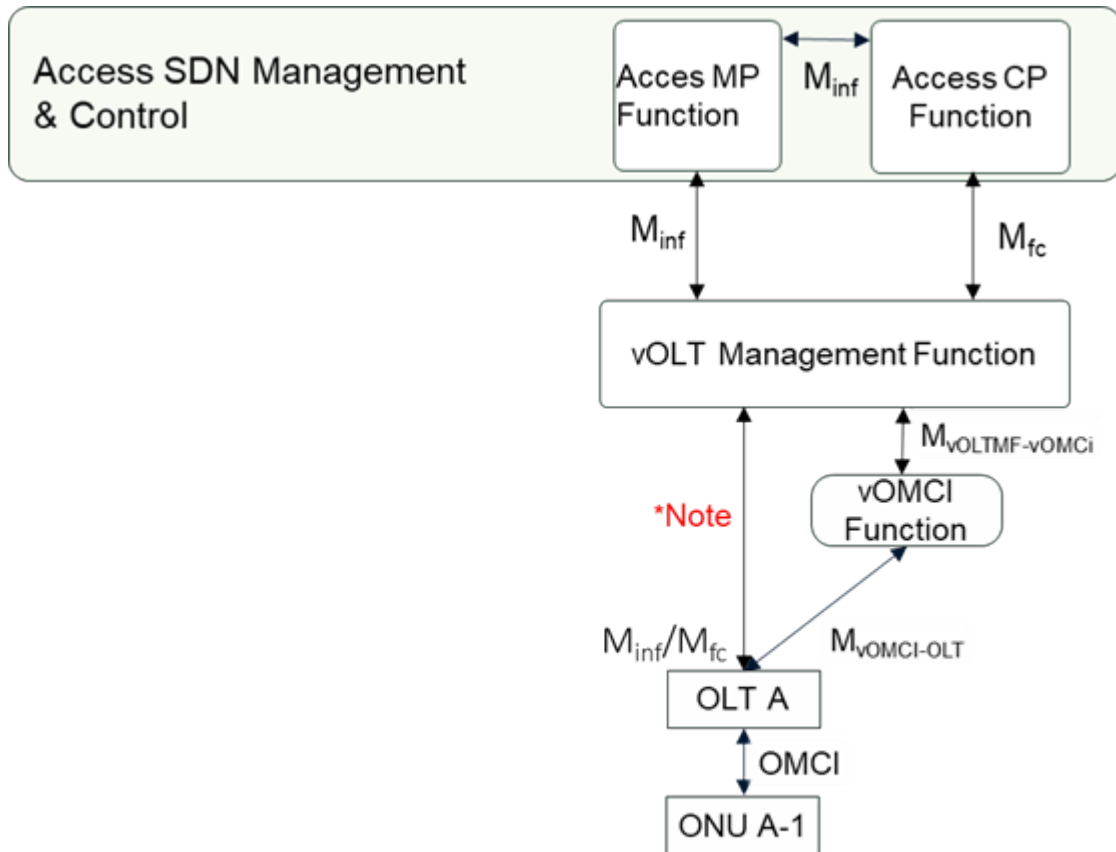


Figure I-2/38 Integrating the vOMCI solution with the CloudCO BAA layer

This way the BAA layer exposes to the northbound Access SDN Management and Control function the same representation and management capabilities for an ONU regardless of the fact the ONU device is embedded-OMCI or virtual-OMCI managed.

I.2 The vOMCI Solution Deployed as Standalone NFVI

In deployment scenarios where the BAA layer is not used in the CloudCO framework, the Access SDN M&C can directly interact with the vOLT Management function. In this deployment scenario the vOLT Management function and the vOMCI function are deployed within an independent Network Function Virtualization Infrastructure (NFVI) as depicted in Figure I-3.



***Note:** The interface between vOLT management and Access PNFs (i.e. OLT) is defined only for functions related to this specification.

Figure I-3/Figure 39 vOMCI Solution Deployed as Standalone NFVI

I.3 CloudCO Framework with vOMCI Solution

The options for deployment of the vOMCI solution in the CloudCO framework as described in I.1 and I.2 is depicted in Figure I-4.

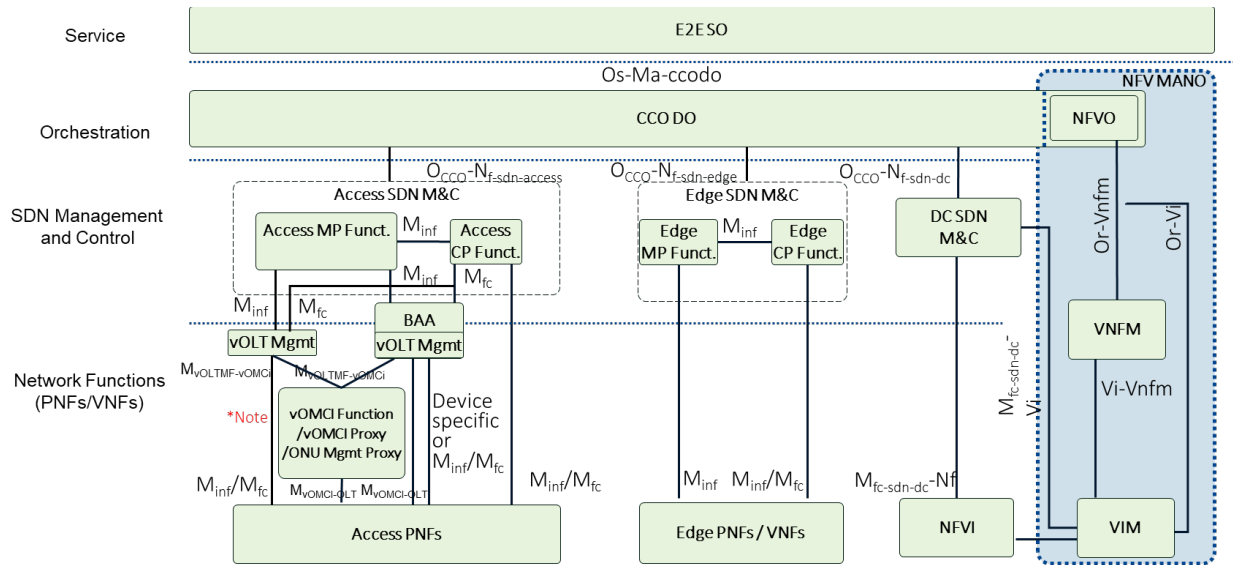
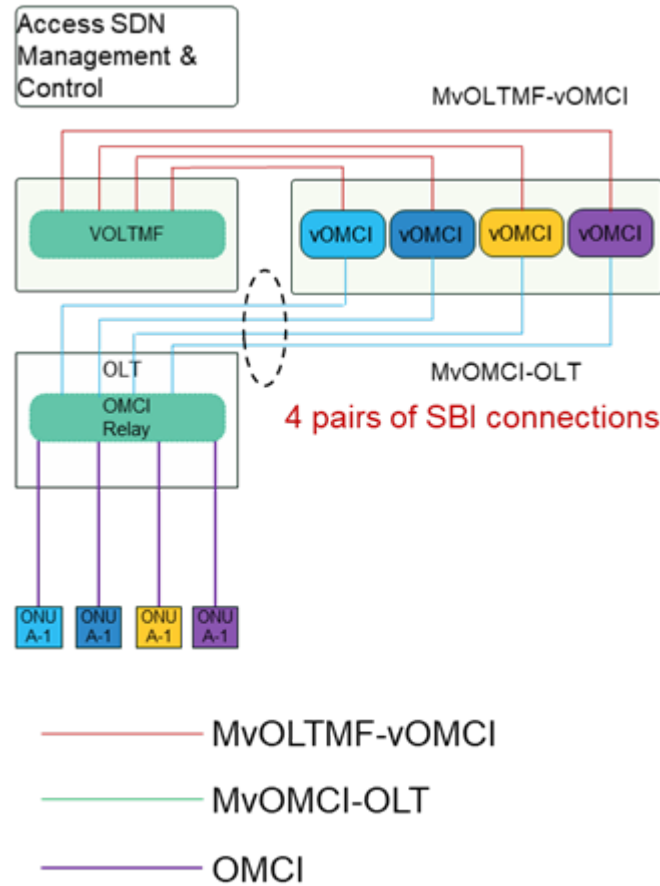


Figure I-4/CloudCO Framework with vOMCI Solution

The vOMCI solution toward the Access PNFs is via the SBI of the vOMCI function and/or vOMCI Proxy as specified in section 5.8 of this Technical Report. Depending on the deployment scenario the vOLT Management function's capabilities as specified in section 5.6 of this Technical Report are provided by either the Access SDN M&C or the BAA layer. Regardless of how the vOMCI function is deployed (e.g., stand-alone, VNF), the capabilities of the vOMCI function and vOMCI Proxy as specified in section 5.2 of this Technical Report and exposed through vOMCI function's NBI remain the same. The only difference is the protocol used to convey the information elements which is specified in section 5.7 of this Technical Report.

I.4 vOLTMF Implemented as an Abstraction Layer between the OLT and vOMCI function

In Appendix I.3, the vOLTMF is either considered part of the BAA layer or as a stand-alone entity as part of the CloudCO. In these previous deployment options, the OLT connects to each instance of the vOMCI function for the ONUs that are attached to the OLT as depicted in the figure below.

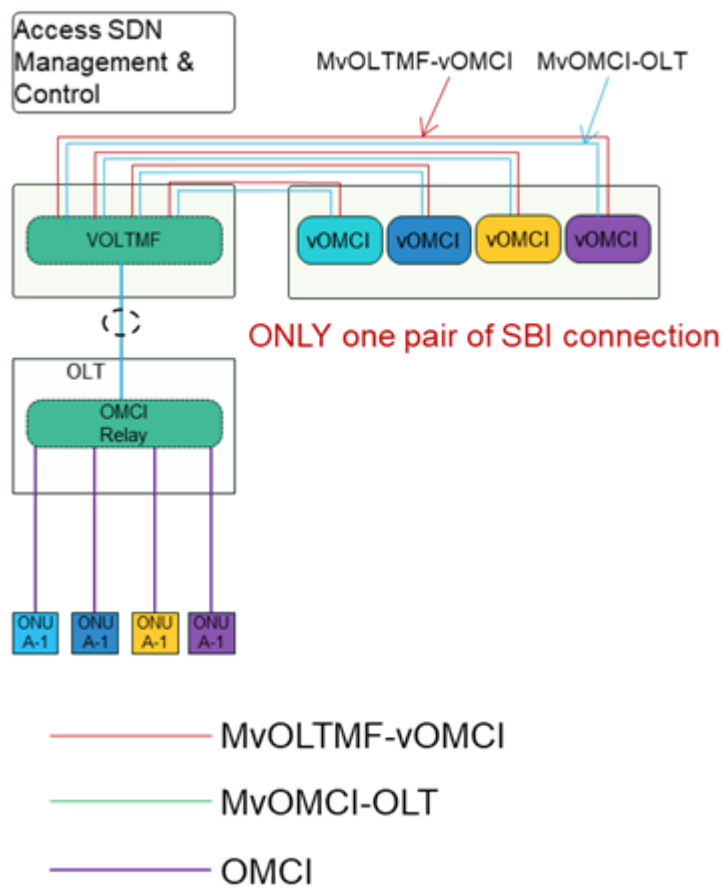


Note: Unless specifically shown, the lines are considered bidirectional

Figure I-5/Figure 41 Example of a OLT with individual sessions to vOMCI functions

An alternative deployment option this section introduces an aggregation of the communication sessions between OLT and one or more vOMCI functions. In doing the aggregation of the communication sessions, the OLT doesn't require connectivity information about each of the vOMCI functions. Instead the OLT only needs connect with the vOMCI Proxy that can be deployed with the vOLTMF to isolate the deployment information of vOMCI function. In doing so, the OLT no longer needs to know how to connect to the vOMCI function which allows the vOMCI function to only need knowledge of the ONU(s) as the service termination point instead of the attached OLT. This alleviates the strict coupling between the OLT and vOMCI function which may expose too much information of vOMCI function instance (e.g., the locations, numbers, capacities, life cycles entities in the vOMCI solution) to parties that are not part of the vOMCI solution. This coupling can also complicate or possibly inhibit the flexible migration of vOMCI functions in a Cloud deployment.

When the vOLTMF is used as the aggregation entity, the vOLTMF has knowledge of the vOMCI solution to include the vOMCI functions, OLTs and ONU attachments and thus is able to easily provide the capabilities to proxy messages between the ONU and vOMCI function via dedicated channels to the requisite OLT. This deployment is depicted below.



Note: Unless specifically shown, the lines are considered bidirectional

Figure I-6/Figure 42 Example of the vOLTMF to Aggregate vOMCI/ONU communication

When the vOMCI function is integrated with the BAA layer as described in appendix I.1 the communication sessions are managed by BAA layer functionality that comprise the vOLTMF as depicted below.

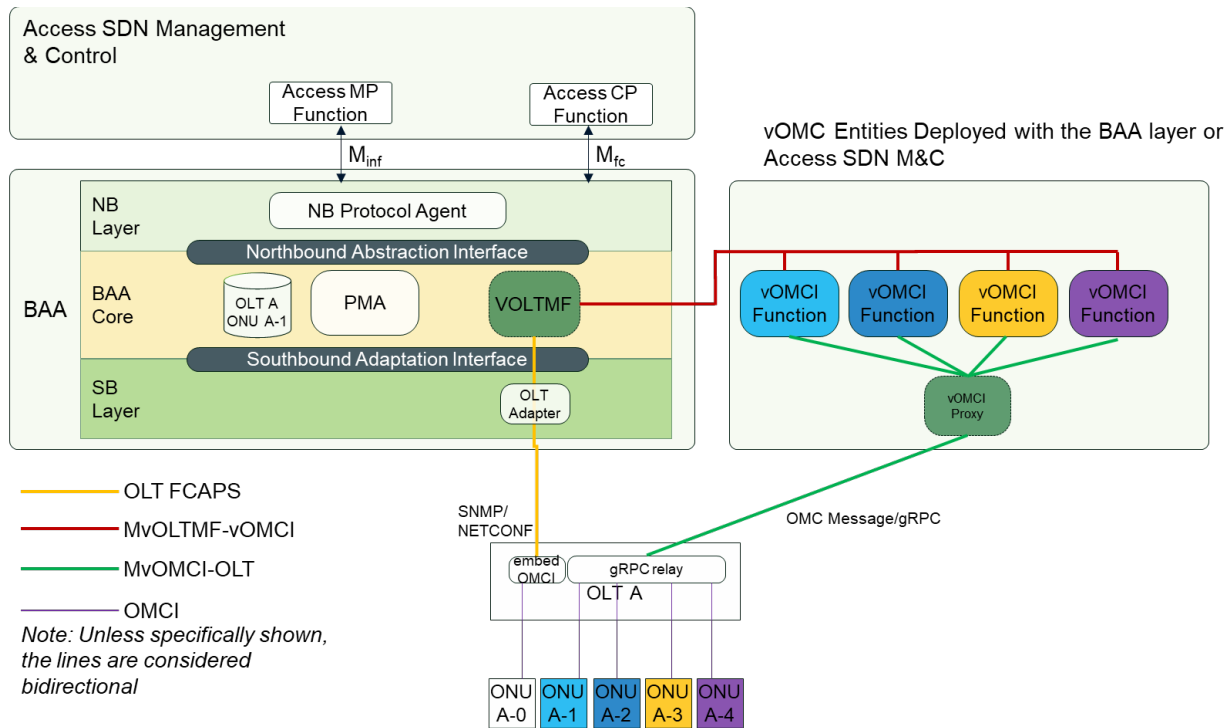


Figure I-7/Figure 43 Deployment vOLTMF, vOMCI function and vOMCI Proxy in the context of the BAA layer

In Figure I-7, the M_{vOLTMF-vOMCI} interfaces are established from the vOLTMF to the vOMCI function. The M_{vOMCI-OLT} interfaces are established between the vOMCI function, vOMCI Proxy and OLT. The vOLTMF maintains the ONU management chain topology for each ONU and establishes the necessary connectivity to the ONU. Only one communication session is necessary to be initiated/maintained between a vOMCI Proxy and OLT. The vOMCI Proxy acts as an application gateway that forwards messages between the vOMCI function and the ONU via the attached OLT.

Appendix II. vOMCI Deployment in FANS

TR-370 [36] Fixed Access Network Sharing (FANS) describes an architectural framework for sharing resources of an Access Node where Infrastructure Network Provider's slices the Access Node's resources that are shared and allocated to virtual network operators (VNOs). TR-370 [36] defines two possible models for resources sharing or slicing:

- Management System based, which performs network slicing at management system level and not directly into the equipment itself. This model is described in Figure II-1.
- Virtual Access Node based, which separates the physical access node into multiple, virtual partitions, each containing ports and forwarding resources directly managed by a VNO. This model is described in Figure II-2.

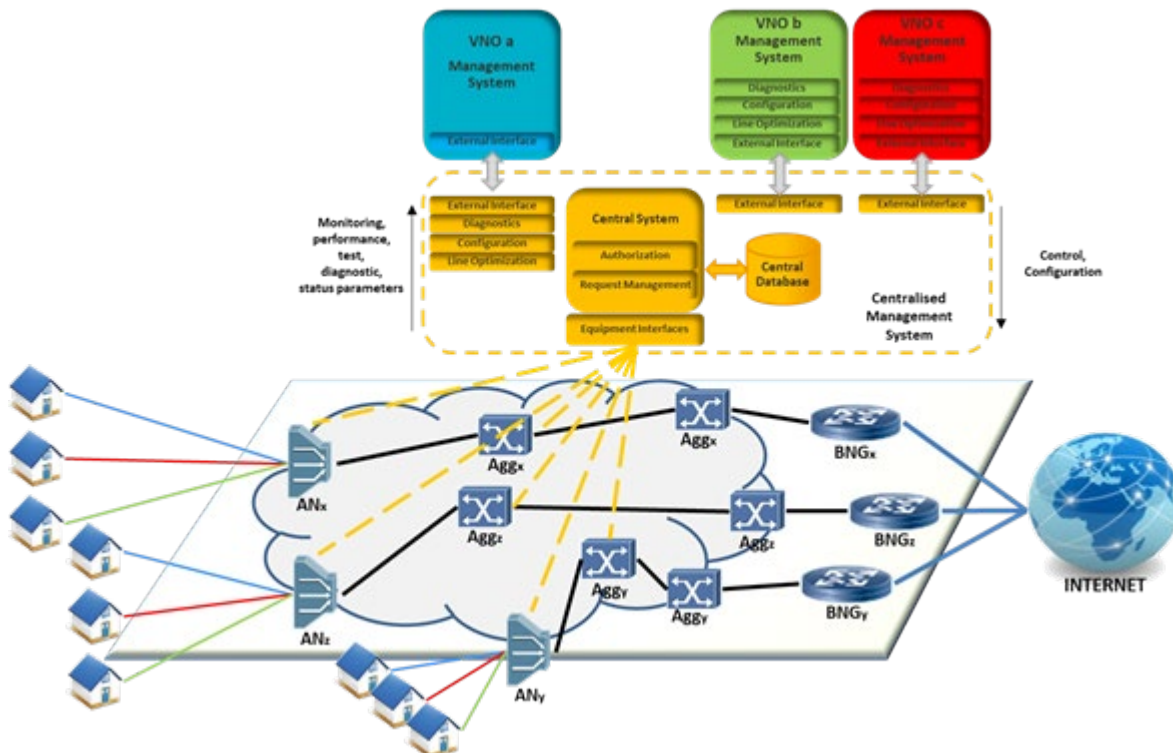


Figure II-1/ Figure 44 TR-370 Management System Model

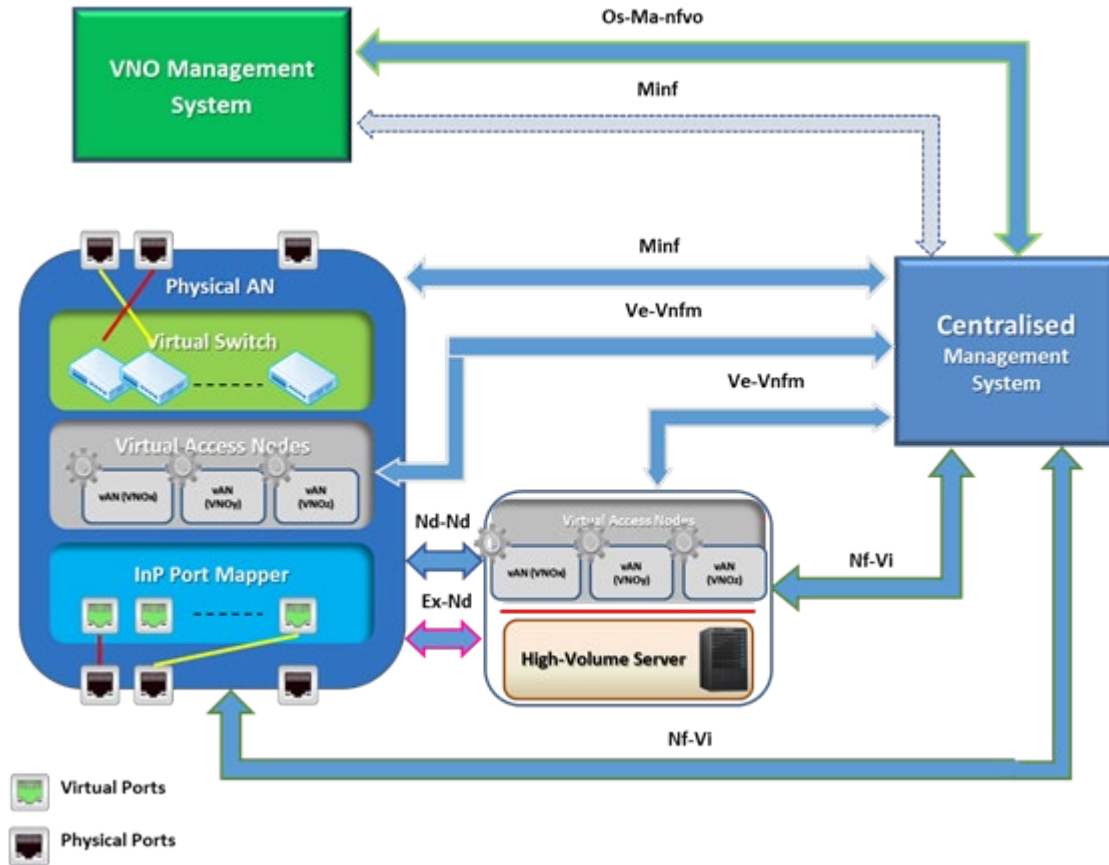


Figure II-2/Figure 45 Virtual Access Node Model

II.1 Management System Model

When FANS is deployed using the Management System Model, the vOLT Management function resides with the Centralized Management System and implements the capabilities defined by the vOLT Management function (e.g., xPON Function, PMAA function, representation of the OLT and ONU's configuration and state) specified in section 5.6 of this Technical Report. As there isn't a Virtualized Access Node in the Management System model, the vOMCI function is a new functional element that can reside in the Centralized Management System or can be introduced with the network as a network function. When deployed with the Centralized Management System, the Centralized Management System provides the capabilities and interfaces provided by the vOMCI function as specified in sections 5.2, 5.7 and 5.8 of this Technical Report. These deployment options are depicted in Figure II-3.

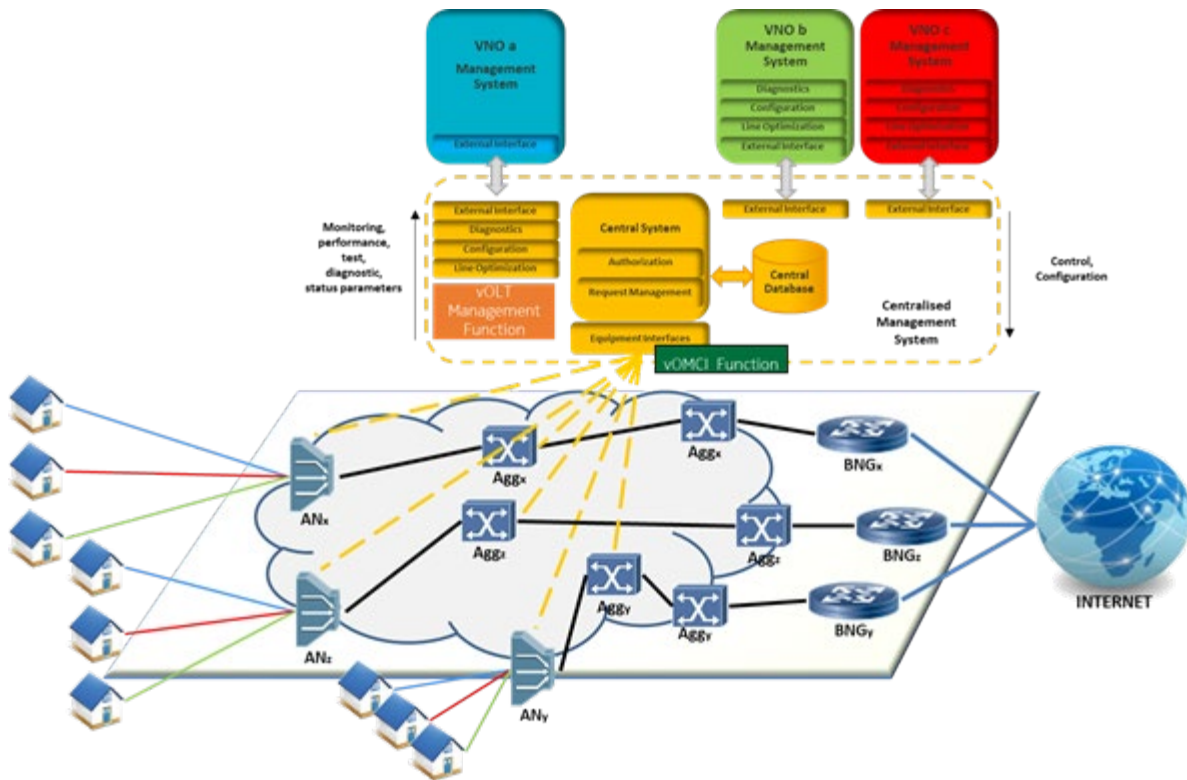


Figure II-3/Figure 46 vOMCI Solution for FANS Management System Model

II.2 Virtual Access Node Model

When FANS is deployed using the Virtual Access Node Model, the vOLT Management function resides on the Centralized Management System implementing the capabilities defined by the vOLT Management function (e.g., xPON Function, PMAA function, representation of the OLT and ONU's configuration and state) specified in section 5.6 of this Technical Report. The NFVI that hosts the Virtualized Access Node provides the capabilities and interfaces provided by the vOMCI function as specified in sections 5.2, 5.7 and 5.8 of this Technical Report or the vOMCI function can reside as a stand-alone function within another NFVI host. This deployment option is depicted in Figure II-4.

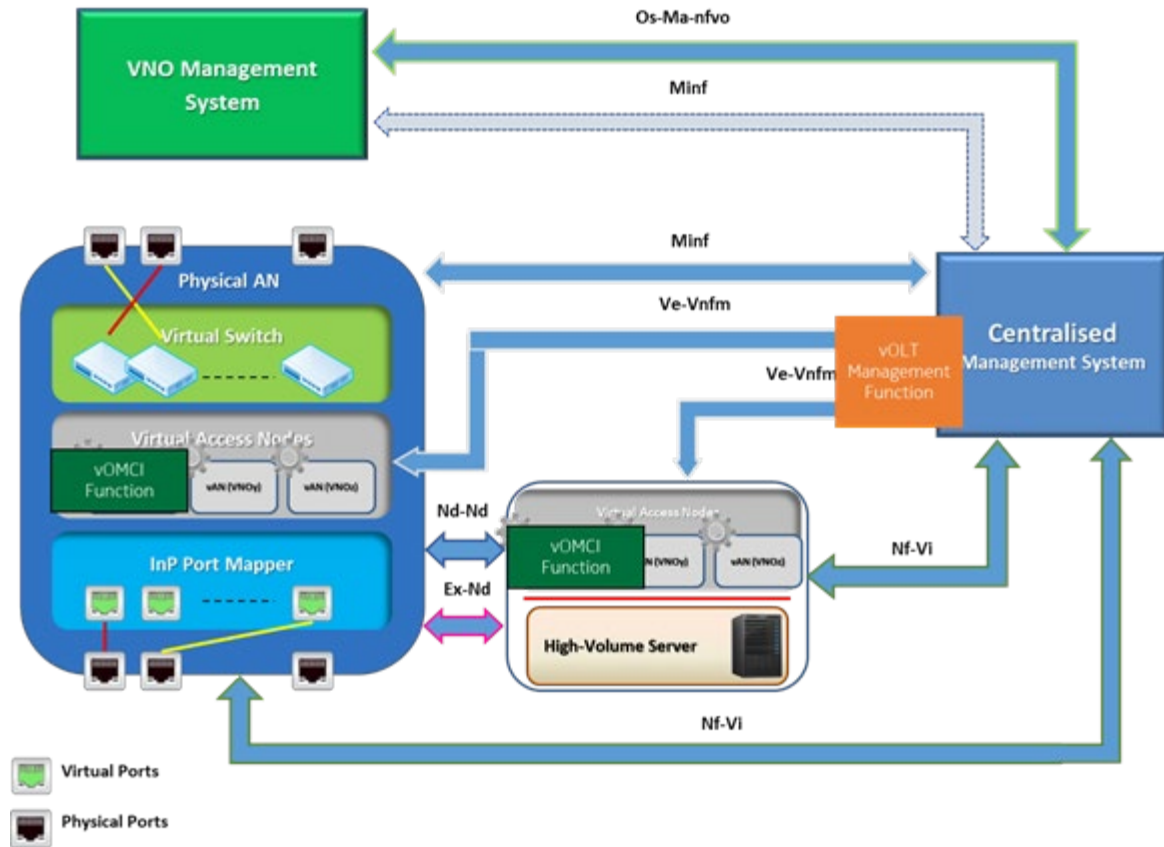


Figure II-4/47 vOMCI Solution for FANS Virtual Access Node Model

Appendix III. Interoperability of vOMCI Functionality with Multi-vendor ONUs

The discovery and management of ONUs in the xPON architecture uses two (2) communication messaging approaches – PLOAM whose messages are communicated at the transmission convergence (TC) layer's PLOAM channel and OMCI messages which are communicated across the OMCI channel. ITU-T G.988 clearly defines the messages and managed entities (ME) but it does not provide guidance on how the MEs are used to support the configuration and operation of ONUs for various services. For example, the deployment of high-speed internet or IPTV services by an operator can use different VLAN architectures that require the use of different MEs in different sequences creating interoperability (IOP) problems when the same services (i.e., IPTV service) are needed across a mix of OLT and ONU vendors. This section describes the problem of multi-vendor interoperability for OMCI messaging in more detail along with two (2) likely deployment scenarios.

III.1 Multi-vendor ONU IOP with OMCI

xPON architectures use OMCI messages to configure and manage functionality provided by the ONU, including service level configuration. In solutions where the OMCI channel is embedded within the OLT, multi-vendor ONU interoperability requires the OLT and subtended ONU establish a common consensus on what OMCI messages to use as well as the sequence needed to fulfil the management request. These MEs and their sequence can differ from ONU or OLT vendor. As such, service providers typically define the OMCI MEs that are used for a service and rely on vendors to determine the proper sequencing of the OMCI messages. These OMCI lists for a service can differ between OLT or ONU vendors and service providers have to spend significant time, using a protocol analyzer, to determine the correct OMCI messages and sequences that are used for the various service interactions. When multiplied by the number of OLT and ONU vendors, the cost and time of this validation becomes prohibitive.

III.2 Multi-vendor ONU IOP with Single vOMCI Function (1:N scenario)

In the 1:N deployment scenario, the service provider is able to define the OMCI ME list for each service and establish the single vOMCI function that can be used with ONUs from different vendors. Because the OLT is not involved in the OMCI message interactions for the service, service providers can verify this single vOMCI function and re-use it to manage all the ONUs. This deployment scenario provides significant reduction in time and cost to validate the solution and reduces the complexity of the ONU IOP with respect to the OLT itself.

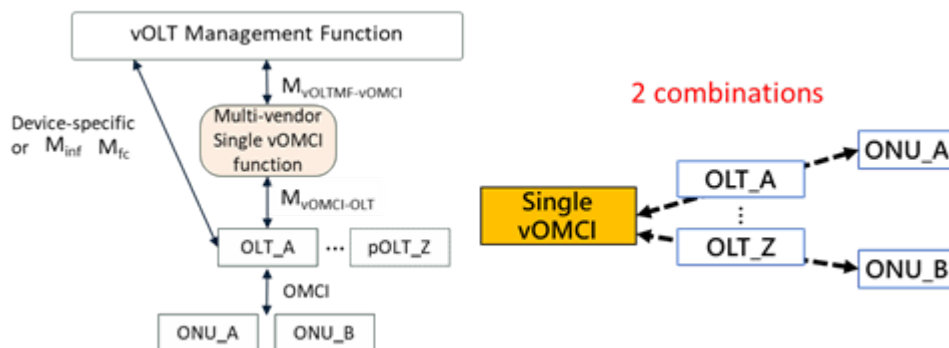


Figure III-1/ Figure 48 ONU IOP with Single vOMCI Function (1:N scenario)

III.3 Multi-vendor ONU IOP with Specific vOMCI Function (N*1:1 scenario)

In the N*1:1 deployment scenario, the OMCI message set or sequencing for the service can vary from ONU vendor or ONU models within a vendor. This deployment scenario requires the capability to group ONUs by various categories like vendor, ONU model or even an ONU device. Each group of ONUs are then able to be associated with a specific vOMCI function. The service provider can reduce the cost to validate the OMCI messages and focus on the service along with the cost of validating the solution.

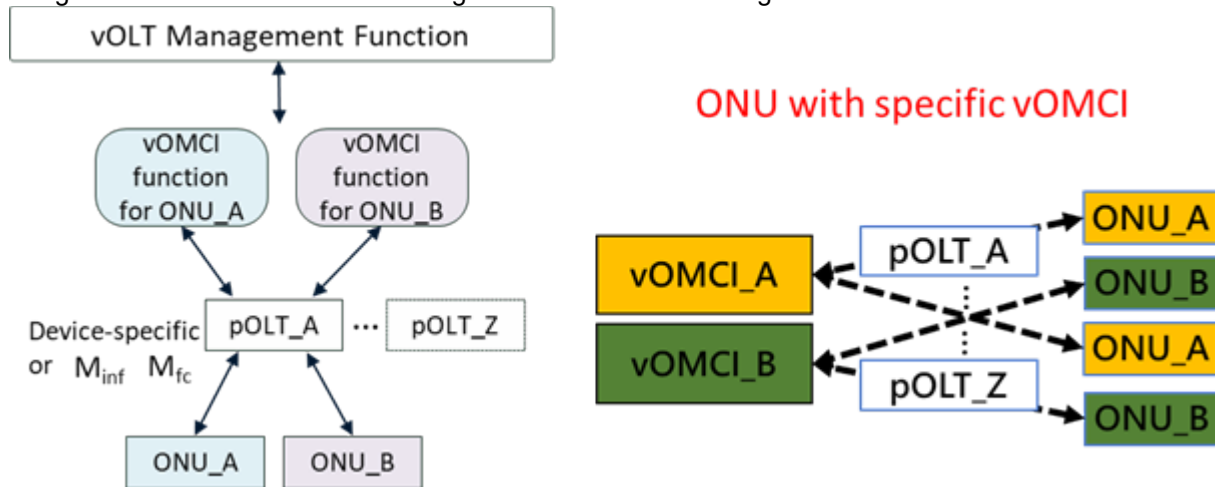


Figure III-2/Figure 49 ONU IOP with Specific vOMCI Functions (N*1:1 scenario)

Appendix IV. ONU Authentication in Various Scenarios

When an ONU attaches to the OLT, the ONU and OLT engage in series of PLOAM interactions that results in the ONU being authenticated as described in ITU-T G.984.3 Appendix VI, ITU-T G.987.3 section 15.2, ITU-T G.989.3 section 15.2 and ITU-T G.9807.1 Annex C.15. Additionally, section 7.1 of TR-156 [29] describes additional methods to authenticate ONUs that can be performed by the OLT or remote management systems (e.g., vOLTMF) when ONUs are initially provisioned.

In addition to the methods discussed in TR-156 section 7.1, ITU-T G.984.3 Appendix VI, ITU-T G.987.3 section 15.2, ITU-T G.989.3 section 15.2 and ITU-T G.9807.1 Annex C.15, there is a need in certain retail scenarios for ONUs and when the ONU can attach to a subset of attachment points within the network (e.g., nomadic ONU) for management systems to be able to validate that the ONU can attach to an attachment point by validating the permitted attachment points that an ONU can be activated upon. This verification check typically is performed based on the allowed topology of attachment points for the ONU.

Further, when an ONU for a subscriber has not been correctly configured for authentication of the ONU (e.g., incorrect attachment point), remote management systems (e.g., vOLTMF) need to be able to authenticate the ONU using information maintained by the remote management system such as a list of valid serial numbers or registration IDs. Additionally, the remote management system can consider the verification of attachment point to avoid unlimited access from unexpected attachment points.

Appendix V. Supporting Interactions for vOMCI Deployments

This appendix defines a series of interactions that are needed to support vOMCI deployments. The purpose of describing these interactions is to identify any requirements that are needed for the interfaces or elements (e.g., OLT, vOLTMF, vOMCI function) of the vOMCI solution.

V.1 Configured ONU is Detected Online

Appendix I of ITU-T G.988 provides a procedure to "bring-up" an ONU that has never been synchronized with the needed managed entities (e.g., vOLTMF). This procedure in ITU-T G.988 is called the "New ONU Bring-up" method.

This section describes the interactions necessary to implement the New ONU bring-up method of Appendix I of ITU-T G.988.

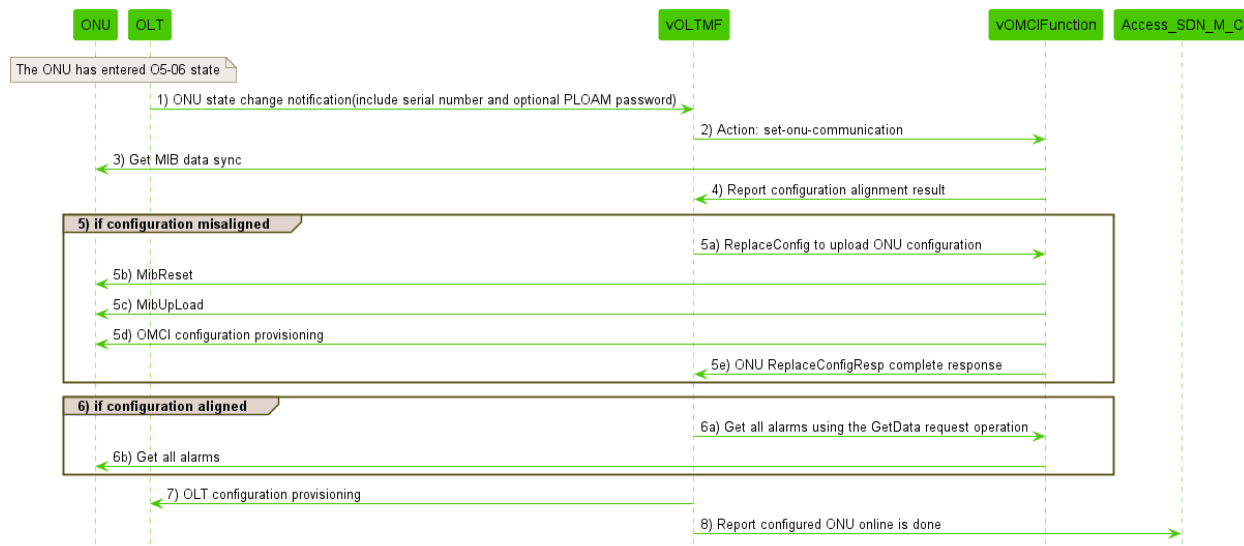


Figure V-1/Figure 50 Configured ONU is Detected Online

The ONU is detected online (OMCC channel has been established) and goes to PLOAM O5-O6 state as defined in Annex D.4/ITU-T G.984.3 [2], Section 12/ITU-T G.987.3 [3], Appendix C.12/ITU-T G.9807.1 [4] and Section 12/ITU-T G.989.3 [5].

1) OLT sends the ONU state change notification (including serial number and optional PLOAM password) to the vOLTMF. The ONU state-change notification is defined in TR-385 issue2 and ITU-T G.984.3 Appendix VI.

2) vOLTMF notifies vOMCI function that an ONU has come online and will use the specified ONU management chain.

3) When the vOMCI function sees an expected ONU has been detected to come online, the vOMCI function sends Get MIB data sync message to ONU as defined in ITU-T G.988 section 9.1.3.

4) vOMCI function reports the configuration align result of the ONU to vOLTMF as defined in ITU-T G.988 Appendix I.1.2.2.

5) If the ONU's configuration is misaligned, the vOLTMF aligns the ONU by sending the ONU's current configuration to the ONU.

5a) vOLTMF sends the ONU's configuration via the ReplaceConfig request message.

5b) vOMCI function issues a MibReset OMCI message as defined in ITU-T G.988 Appendix I.1.4.2.

5c) vOMCI function issues a MibUpload OMCI message as defined in ITU-T G.988 Appendix I.1.1.1.2.

5d) vOMCI function issues any additional configuration necessary to complete the ONU alignment request.

5e) vOMCI function sends a ReplaceConfigResp response message to the vOLTMF with the result of the operation.

6) If the ONU's configuration is aligned, the vOLTMF requests the alarms from an ONU.

6a) vOLTMF sends a request for the alarms via the GetData request message as defined in section 5.7.1 of this Technical Report.

6b) vOMCI function issues a "Get all alarms OMCI message as defined in ITU-T G.988 Section 11.2.2

7) The vOLTMF sends the OLT's configuration to the OLT.

8) If the configuration is aligned, the vOLTMF reports that ONU is detected and aligned to the Access SDN M_C.

V.2 Automated ONU Discovery

When a ONU is attached to an OLT the ONU identifies itself to the OLT via its serial number or registration ID. Upon recognition of the ONU the OLT informs the vOLTMF of the discovered ONU.

In this scenario, the ONU has not been properly authenticated requiring the vOLTMF to request additional ONU device information from the vOMCI. Upon receiving the device information of the ONU via the vOMCI function, the vOLTMF reports the discovery of the ONU to other interested parties (e.g., Access SDN M&C).

Note: If the ONU passes authentication, the flow is shown in section Configured ONU is detected Online.

The figure below describes this interaction.

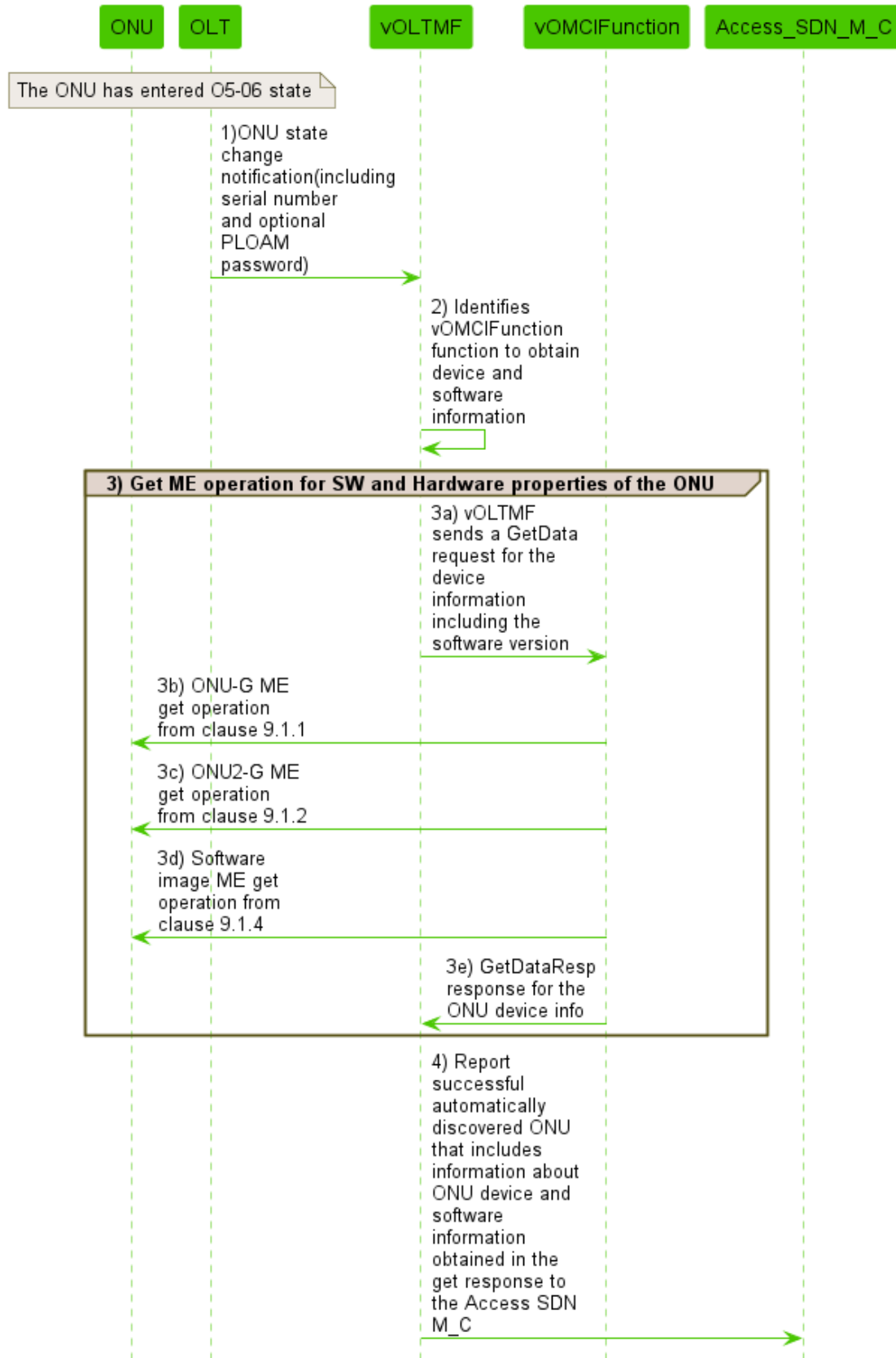


Figure V-2/Figure 51 Automated ONU Discovery

The ONU is detected online (OMCC channel has been established) and goes to PLOAM O5-O6 state as defined in Annex D.4/ITU-T G.984.3 [2], Section 12/ITU-T G.987.3 [3], Appendix C.12/ITU-T G.9807.1 [4] and Section 12/ITU-T G.989.3 [5].

1) OLT sends the ONU state change notification (including serial number and optional PLOAM password) to the vOLTMF. The ONU state change notification is defined in TR-385 issue2 and ITU-T G.984.3 Appendix VI.

2) vOLTMF identifies the vOMCI function that can obtain the device and software information for the ONU.

3) Get ME operation for software and hardware properties of the ONU

3a) vOLTMF sends a request to get the device information including the software version.

3b) vOMCI function sends the ONU-G ME get operation defined in ITU-T G.988 clause 9.1.1.

3c) vOMCI function sends the ONU2-G ME get operation defined in ITU-T G.988 clause 9.1.2.

3d) vOMCI function sends the Software image ME get operation defined in ITU-T G.988 clause 9.1.4.

3e) vOMCI function returns the response for getting ONU device information to the vOLTMF.

4) Report automatically discovered ONU that includes information about ONU device and software information obtained in the get response to the Access SDN M_C.

V.3 Automated ONU Discovery Failure

When an ONU is attached to an OLT the ONU identifies itself to the OLT via its serial number. Upon recognition of the ONU the OLT informs the vOLTMF of the discovered ONU.

In this scenario, the vOLTMF cannot either discern which vOMCI function instance can be used to send the request to obtain the device and software information from the ONU or the selected vOMCI function cannot successfully obtain the device and software information.

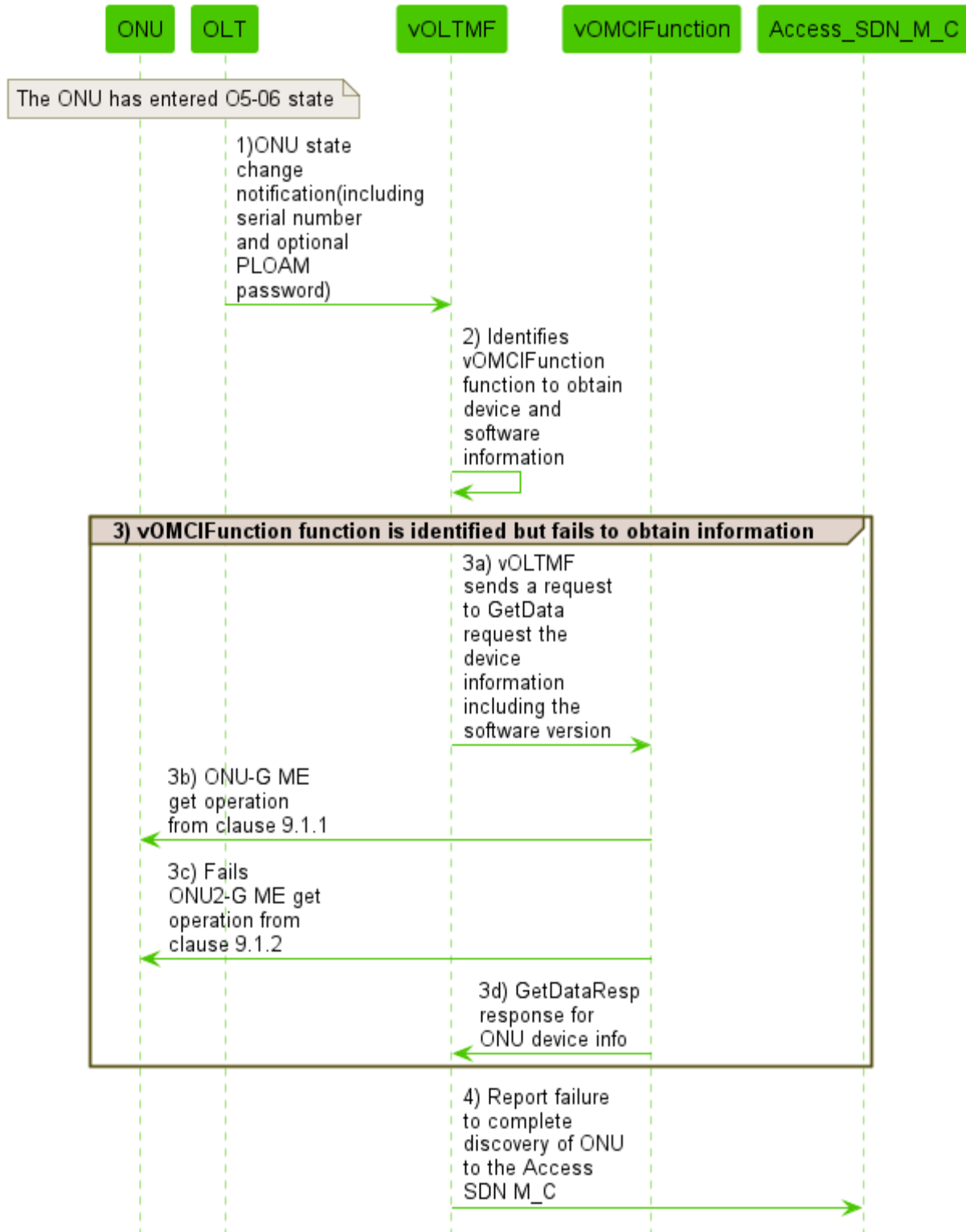


Figure V-3/Figure 52 Automated ONU Discovery Failure

The ONU is detected online (OMCC channel has been established) and goes to PLOAM O5-O6 state as defined in Annex D.4/ITU-T G.984.3 [2], Section 12/ITU-T G.987.3 [3], Appendix C.12/ITU-T G.9807.1 [4] and Section 12/ITU-T G.989.3 [5].

- 1) OLT sends the ONU state change notification (including serial number and optional PLOAM password) to the vOLTMF. The ONU state change notification is defined in TR-385 issue2 and ITU-T G.984.3 Appendix VI.
- 2) vOLTMF identifies the vOMCI function that can obtain the device and software information for the ONU.
- 3) If the vOMCI function is identified, the vOLTMF sends Get ME operation for software and hardware properties of the ONU.
 - 3a) vOLTMF sends a request to get the device information including the software version.
 - 3b) vOMCI function sends the ONU-G ME get operation defined in ITU-T G.988 clause 9.1.1.
 - 3c) vOMCI function fails to receive the ONU2-G ME get operation defined in ITU-T G.988 clause 9.1.2.
 - 3d) vOMCI function returns the failed response for getting ONU device information to the vOLTMF.
- 4) Report failed automatically discovered ONU that includes additional information about the failure to the Access SDN M_C.

V.4 Synchronize PM-related intervals and timer

The ONUs that are managed in the vOMCI solution can be aligned to a common 15-min collection interval by the vOMCI function. The ONUs' PM interval is periodically synchronized with the corresponding PM interval maintained by the vOMCI. This is achieved by utilizing the OMCI synchronize time action to align individual ONUs when necessary (e.g., newly detected, alignment/realignment). Furthermore, the instances of the vOMCI function should also be synchronized to a common clock via the NTP protocol. Thus, ONUs managed by separate vOMCI functions can also be synchronized between each other with rather high precision or accuracy provided by the OMCI synchronize time action.

Section I.4 of ITU-T G.988 [1] describes the capability where PM MEs can be synchronized to a common 15-min collection interval starting point by sending a request to the ONU to synchronize the PM collection interval using the OMCI synchronize time action as described in appendix A.2.33 and A.3.33 of ITU-T G.988 [1].

An example scenario, the Figure V-4 describes the PM 15-min collection interval alignment activity upon discovery on an ONU by adding an additional step in the discovery of the ONU as depicted in Figure V-1:

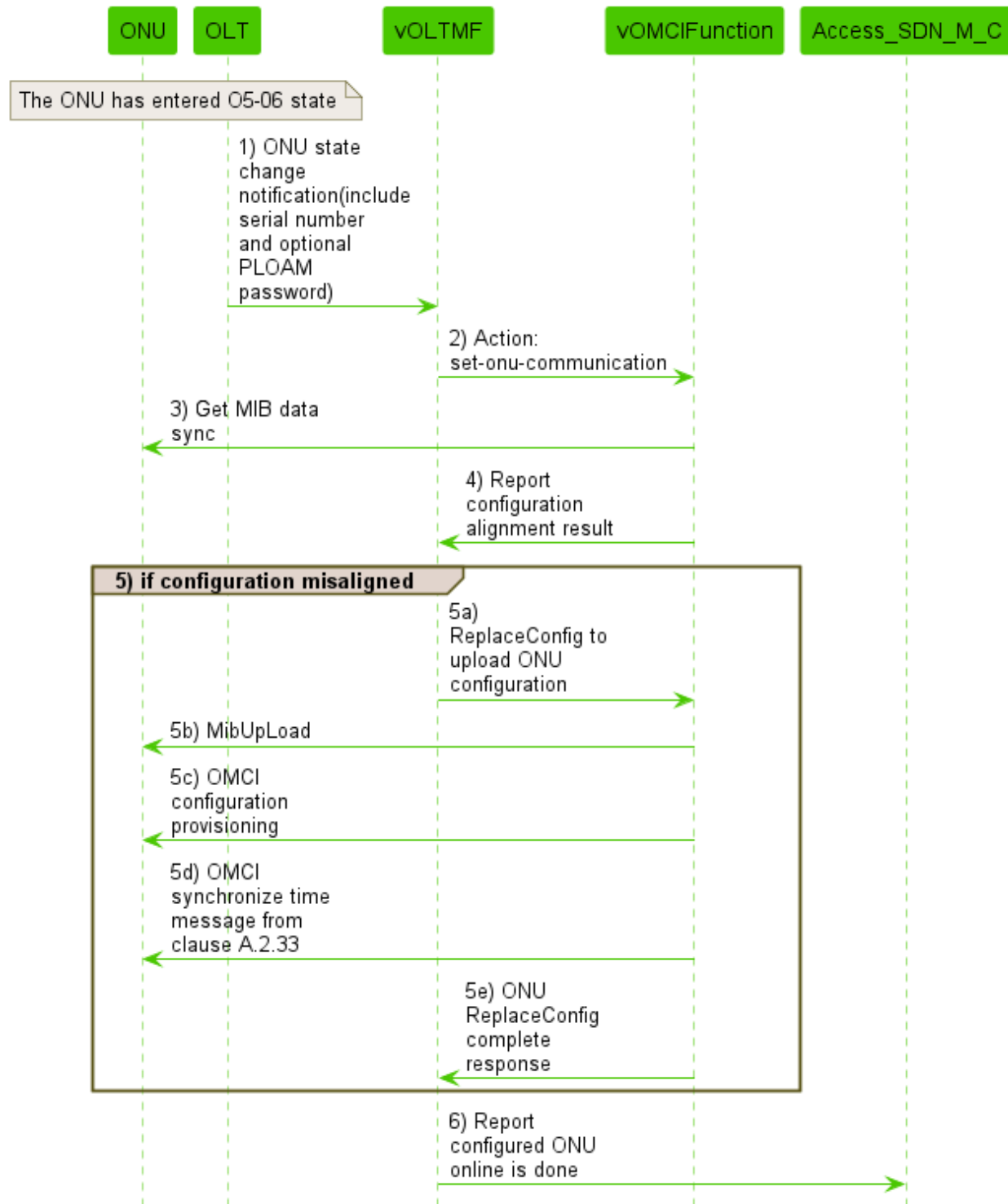


Figure V-4/Figure 53 Synchronize PM intervals for a Discovered ONU

V.5 Background ONU Software Download Support

Typically, ONU Software download via OMCI is a prolonged process, especially if baseline OMCI messages are used for image transfer. The vOMCI function can identify whether ONU supports extended OMCI messages by querying ONU2-G ME, as defined in ITU-T G.988 clause 9.1.2 and use extended OMCI messages if supported by the ONU.

Since image transfer is lengthy and involves multiple “Download section”, “Download section response” iterations, the vOMCI function must allow other configuration interactions while image transfer is in progress in the background. Following software download, the new software image can optionally be committed and/or activated. If software image is activated, ONU restarts and then the ONU is discovered as described appendix V.2.

The background ONU software download example is depicted in Figure V-5:

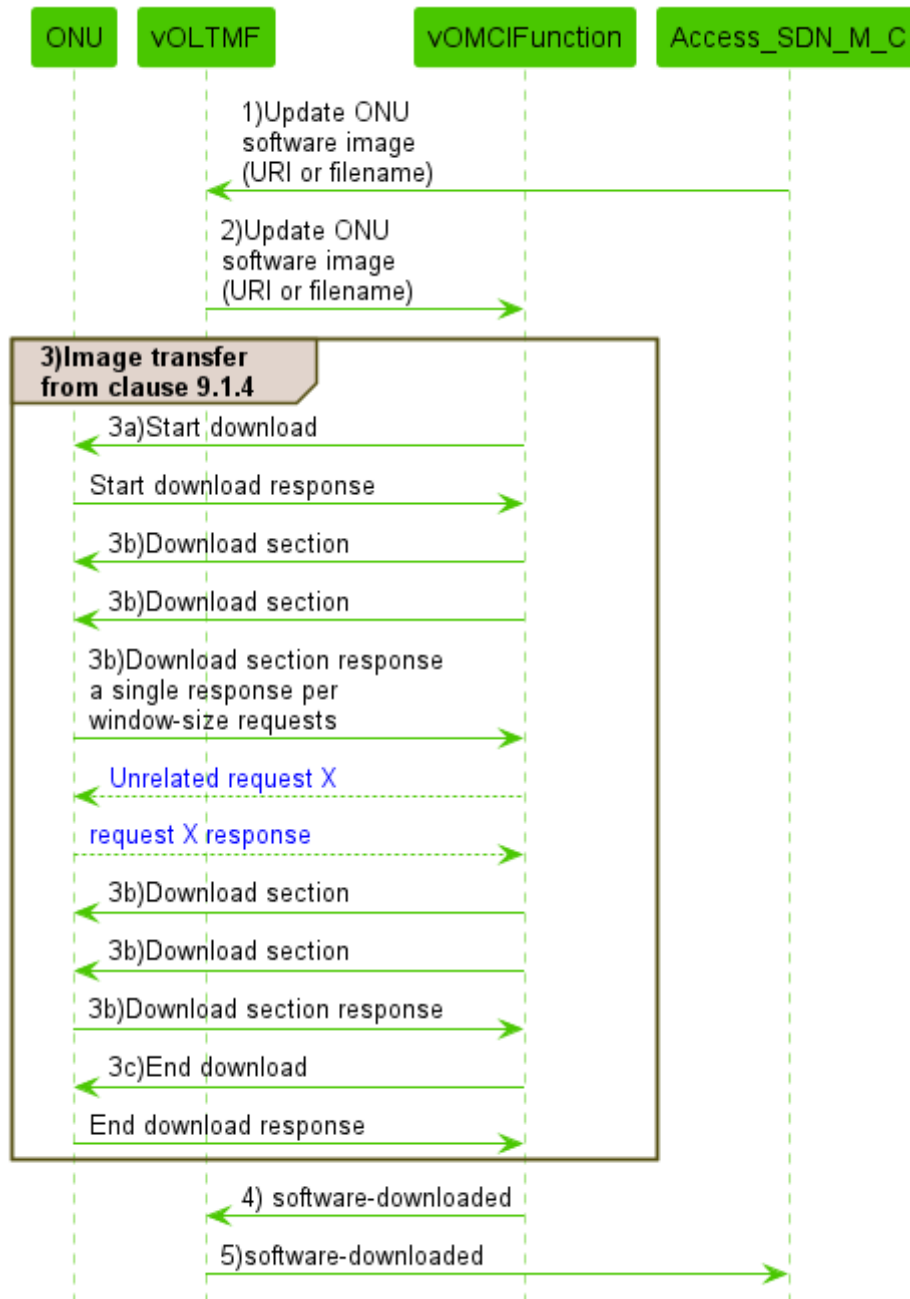


Figure V-5/Figure 54 Background ONU Software Download

1) Access SDN M_C requests vOLTMF to download a new ONU software image using download” action, as defined in Common YANG, bbf-software-management.yang.

2) vOLTMF forwards the “download” action to vOMCI function instance managing the target ONU.

3) vOMCI function transfers software image using “Start download” operation, followed by multiple “Download Section” operations”, followed by “End Download” operation. All operations are applied to “Software Image” ME, as defined in ITU-T G.988 Clause 9.1.4.

ONU image transfer typically takes from multiple seconds to multiple minutes. vOMCI function must be able to modify and query ONU configuration whilst image transfer is in progress. In the diagram above it is shown as “Unrelated request X” in the step 3b.

4) vOMCI function notifies vOLTMF that software download is completed using revision-downloaded notification, as defined in the TR-383 [33] YANG module, `bbf-software-management.yang`.

5) vOLTMF notifies Access SDN M_C.

Appendix VI. Connectivity Management among vOLTMF(s) and vOMCI functions

In Figure VI-1 one or more vOLTMF instances manage the connectivity and interaction with multiple vOMCI function instances (i.e., vOMCI1, vOMCIn). Each vOLTMF instance could interact with one or more vOMCI function instances directly or indirectly, e.g., via load balancing. The vOLTMF is responsible for the workflow and sequencing the requests to the ONU. Each vOLTMF instance can perform parallel configurations of multiple different ONUs. Moreover, the vOLTMF takes the responsibility of maintaining the management and control states of the ONUs and makes the ONU management and control complete and accurate.

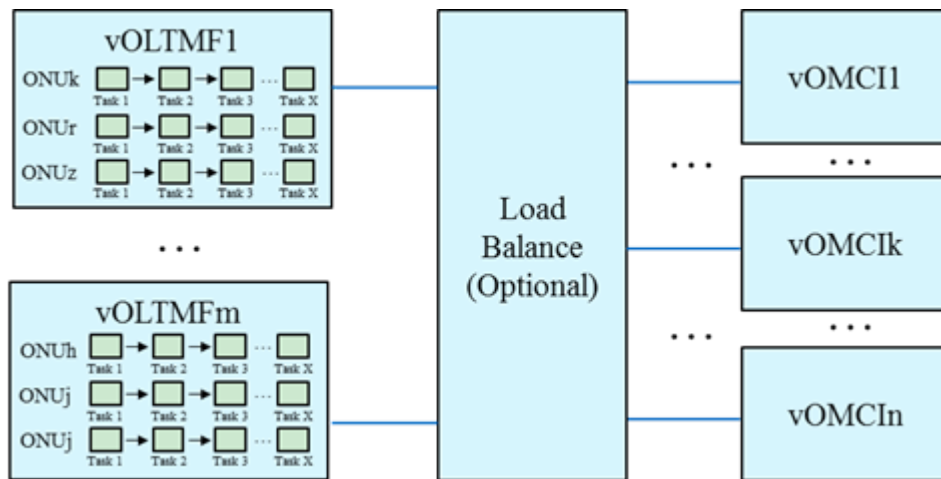


Figure VI-1/Figure 55 Connectivity Among vOMCI functions and vOLTMF(s)

Appendix VII. vOMCI Function Connectivity Using a Session Proxy

In certain deployments the OLT can consolidate the communication with N vOMCI functions using the same communication session to a communication end-point that acts as a transparent session proxy. In Figure 5.5-1, the OLT A uses 2 communication sessions to the vOMCI function instances. However, if a single session from OLT A is required, the vOMCI Proxy can be used. This deployment scenario works well where the OLT is resource constrained.

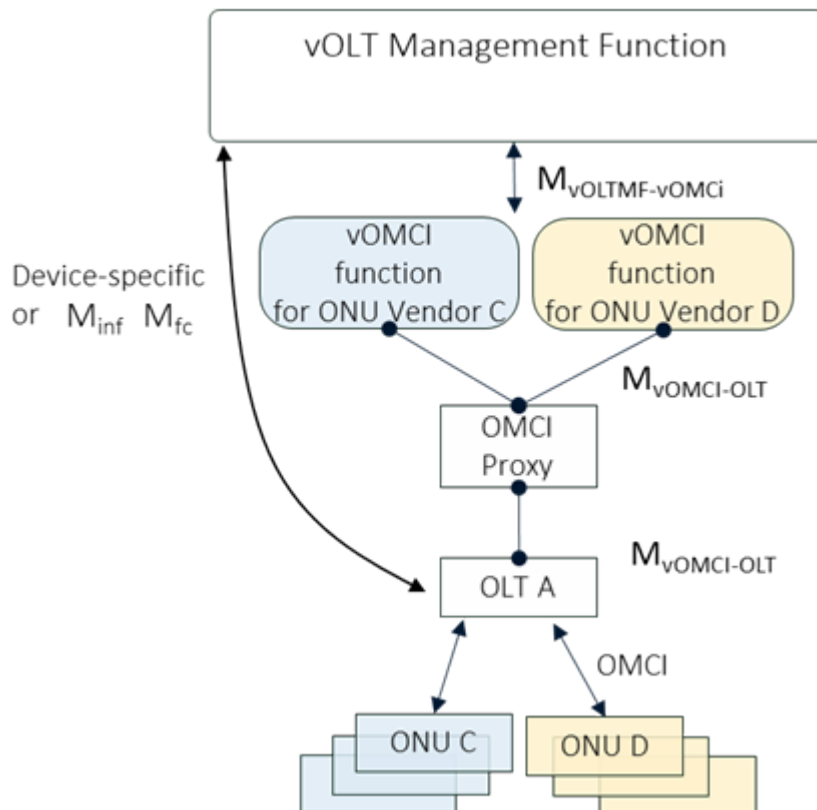


Figure VII-1/Figure 56 Single Session – OLT to vOMCI Connectivity

Appendix VIII. Examples of M_{vOLTMF-vOMCI} Messages

Section 5.7.1 provides the information model and message definition for messages that are conveyed across the M_{vOLTMF-vOMCI} interface. This appendix provides examples for usage of the various requests that use the M_{vOLTMF-vOMCI} interface.

VIII.1 ONU Device Data Model

Section 5.7.1 describe the message format that is used to convey the data model and operations to configure and operate an ONU device. This section describes examples of the *ReplaceConfig* request message and an example *Notification* for ONU alignment.

VIII.1.1 Example ReplaceConfig Message for ONU Target

The following is an example of a *ReplaceConfig* request to configure onu1:

ReplaceConfig

```
Msg {
  header {
    msg_id: "2"
    sender_name: "vOLTMF"
    recipient_name: "vomci-vendor-1"
    object_type: ONU
    object_name: "ont1"
  }
  body {
    request {
      replace_config {
        config_inst: "{\"bbf-qos-filters:filters\":{},\"ietf-ipfix-psamp:ipfix\":{},\"ietf-hardware:hardware\":{},\"bbf-qos-classifiers:classifiers\":{},\"ietf-pseudowires:pseudowires\":{},\"bbf-ghn:ghn\":{},\"bbf-qos-policies:qos-policy-profiles\":{},\"bbf-vdsl:vdsl\":{},\"bbf-xpongmtcont:xpongmtcont\":{},\"ietf-system:system\":{},\"bbf-qos-policer-envelope-profiles:envelope-profiles\":{},\"bbf-ghs:ghs\":{},\"ietf-interfaces:interfaces\":{},\"bbf-qos-policies:policies\":{},\"bbf-qos-traffic-mngt:tm-profiles\":{},\"bbf-qos-policing:policing-profiles\":{},\"bbf-selt:selt\":{},\"bbf-l2-dhcpv4-relay:l2-dhcpv4-relay-profiles\":{},\"ieee802-dot1x:nid-group\":{},\"ietf-alarms:alarms\":{},\"ietf-netconf-acm:nacm\":{},\"bbf-hardware-rpf-dpu:rpf\":{},\"bbf-pppoe-intermediate-agent:pppoe-profiles\":{},\"bbf-fast:fast\":{},\"bbf-mgmd:multicast\":{},\"bbf-melt:melt\":{},\"bbf-subscriber-profiles:subscriber-profiles\":{},\"bbf-ldra:dhcpv6-ldra-profiles\":{},\"bbf-l2-forwarding:forwarding\":{}}"
      }
    }
  }
}
```

ReplaceConfigResp response

```

Msg {
  header {
    msg_id: "2"
    sender_name: "vomci-vendor-1"
    recipient_name: "vOLTMF"
    object_type: ONU
    object_name: "ont1"
  }
  body {
    response {
      replace_config_resp {
        status_resp {
          status_code=0
        }
      }
    }
  }
}

```

VIII.1.2 Example Action Message to Set and ONU Management Chain

The following is an example of an *Action* message that is sent from the vOLTMF to a vOMCI function in order to set an ONU's management chain.

Action

```

Msg{
header {
  msg_id: "3"
  sender_name: "vOLTMF"
  recipient_name: "vomci-vendor-1"
  object_type: VOMCI_FUNCTION
  object_name: "vomci-vendor-1"
}
body {
  request {
    action {
      input_data: "{\"bbf-vomci-function:managed-onus\":{\"managed-onu\":{\"name\":\"ont1\",\"set-onu-communication\":{\"onu-attachment-point\":{\"olt-name\":\"OLT1\",\"channel-termination-name\":\"CT_1\",\"onu-id\":1},\"voltmf-remote-endpoint\":\"vOLTMF_Kafka\",\"onu-communication-available\":true,\"olt-remote-endpoint\":\"proxy-grpc-1\"}}}}}"
    }
  }
}

```

```
}  
}
```

Action response

```
Msg {  
  header {  
    msg_id: "3"  
    sender_name: "vomci-vendor-1"  
    recipient_name: "vOLTMF"  
    object_type: VOMCI_FUNCTION  
    object_name: "vomci-vendor-1"  
  }  
  body {  
    response {  
      action_resp {  
        status_resp {  
          status_code=0  
        }  
      }  
    }  
  }  
}
```

Appendix IX. Examples for Using Datastore Tag

Section 5.6.4.1 discusses that DSTs can be used between the vOLTMF and vOMCI function in order keep the ONU configuration datastore maintained by the vOLTMF with the ONU datastore information kept by the vOMCI function. This section describes several examples that depict the usage of the DST.

IX.1 Use of DST in Configuration Requests

The vOLTMF can use the datastore tag to ensure the latest ONU configuration known by the vOMCI function is the latest ONU configuration known by the vOLTMF. The vOLTMF does this by communicating the latest datastore tag value to the vOMCI function in configuration requests as depicted in Figure IX-1. In this figure the vOLTMF sends a different value for the DST each time the vOLTMF makes an update to the ONU's configuration data. The vOMCI function maintains the latest successful provisioning/configuration request in addition to maintaining the OMCI MIB Data Sync value received from the ONU. The vOLTMF can then determine if the ONU configuration maintained by the vOLTMF and vOMCI function is synchronized by comparing the value of the DST retrieved using the GetData message.

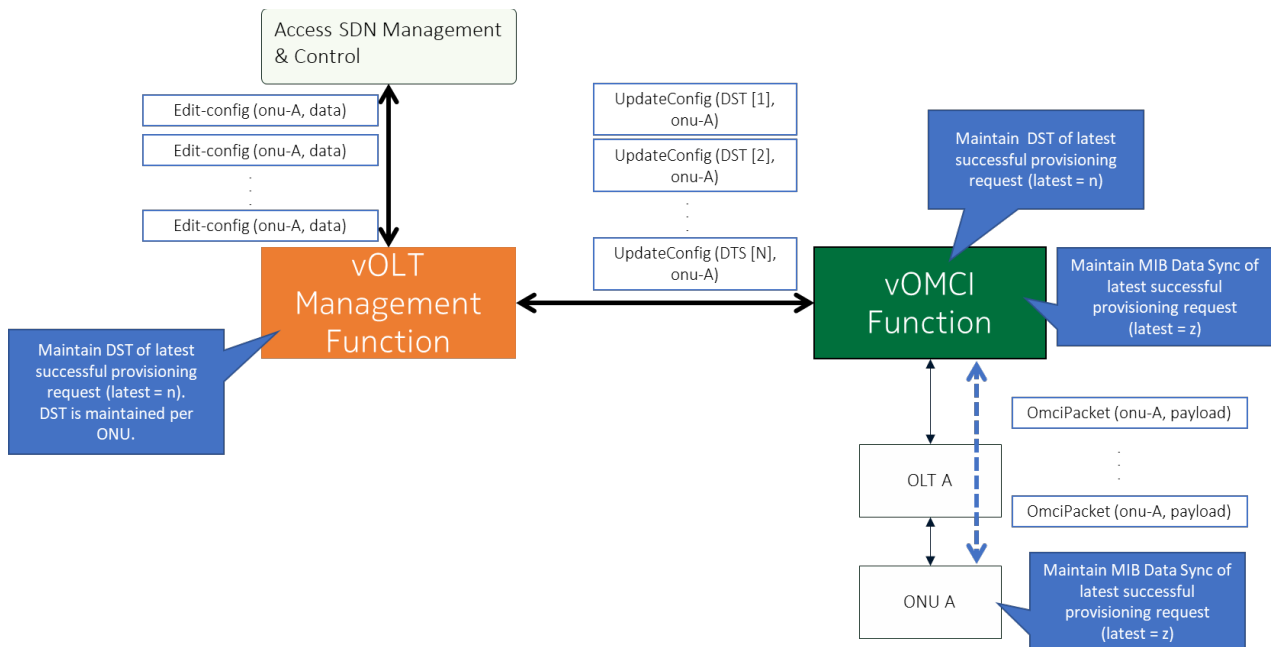
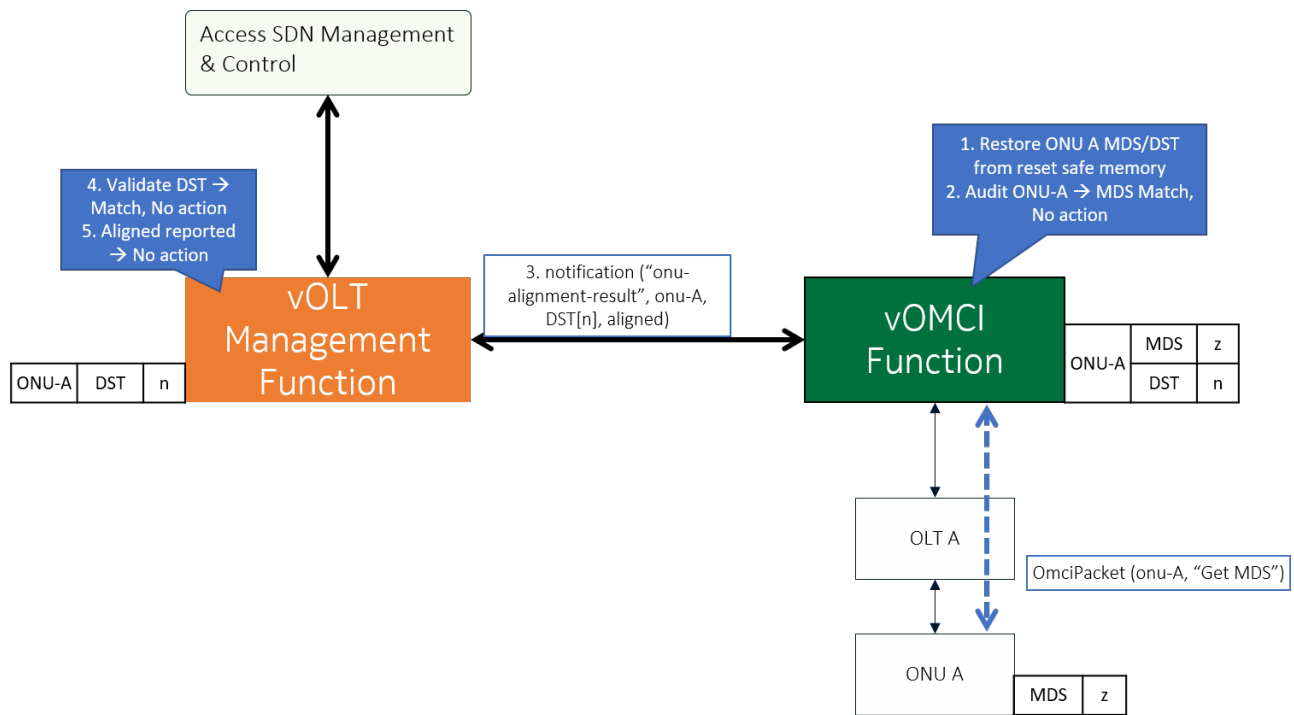


Figure IX-1/57 Use of DST in ONU Configuration

IX.2 Use of DST when the vOMCI function has Restarted

When a vOMCI function is restarted, the vOLTMF can use the DST to quickly determine if the vOMCI function's ONU configuration is aligned with the vOLTMF ONU configuration. The use of the DST along with the vOMCI function's use of the MDS provides an efficient and fast mechanism to ensure the configuration information for an ONU in the ONU, vOMCI function and vOLTMF are aligned. In Figure IX-2, when the vOMCI function is restarted, the vOMCI function aligns itself with the ONU using the MDS and then reports the DST in the alignment notification where the vOLTMF can then use to determine if the DST reported by the vOMCI function is the same as the DST value maintained by the vOLTMF. Between the alignment result and the DST value the vOLTMF can determine if the ONU needs to have its configuration restored.

**Figure IX-2/Figure 58 Use of DST in vOMCI Function Restart**

Appendix X. vOMCI-as-a-Service Deployment

The vOMCI-as-a-Service deployment organizes the ONU Management Proxy, vOMCI Proxy and vOMCI function entities defined in this Technical Report into a vOMCI service that is consumed by multiple client instances within the vOLTMF that is in charge of the various management functionalities such as ONU access and slice management as depicted in Figure X-1 below:

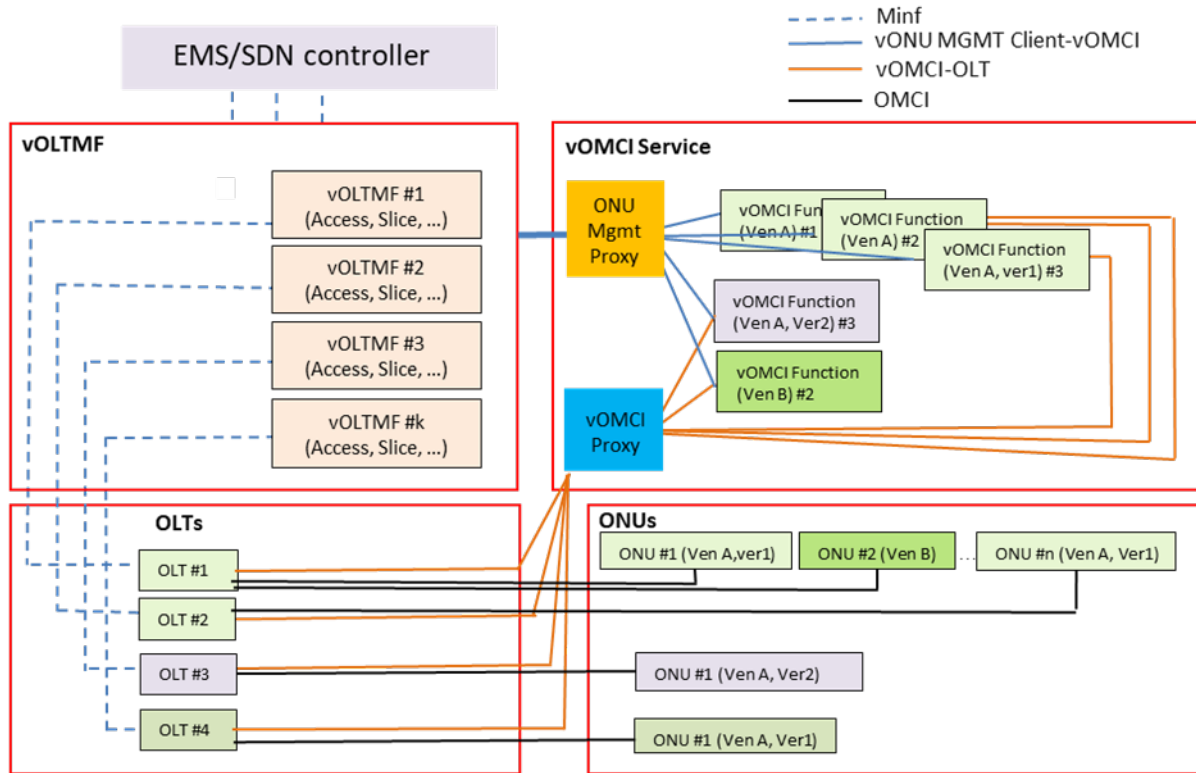


Figure X-1/59 vOMCI-as-a-Service

By organizing the ONU Management Proxy, vOMCI Proxy and the associated vOMCI function instances as a service and disaggregating from the vOLTMF the vOMCI service logic such as association of ONUs and vOMCI function instances, ONU management chain setup, manipulation of the load balance and scale in/out of the vOMCI function instances, the complexity of the connectivity between entities (e.g., vOLTMF instance, OLT) that interface with the service is greatly reduced as the vOLTMF instances and OLTs no longer need to connect directly to each vOMCI function instances. Instead the OLTs would connect to a vOMCI Proxy and the vOLTMF instance connects to the ONU Management Proxy. This simplification has the following benefits:

- When the scaling behavior of the vOMCI function instances rapidly changes (e.g., during restoration of a geographic outage) and/or the number of vOMCI function instances needed to manage the ONUs is large, the vOLTMF and OLT instances are no longer affected instead the burden of interactions toward the vOMCI function instances is contained within the vOMCI service and its ONU Management Proxy and vOMCI Proxy entities.
- Simplification of the clients consuming vOMCI services such as vOLTMF instances is accomplished by relying on the ONU Management Proxy to maintain the connectivity to the vOMCI function instances and also provide the capability of allocating ONUs to vOMCI function instances. The distribution of this functionality to allocate, monitor and maintain the association between the ONU

and vOMCI function instance to the vOMCI Service's ONU Management Proxy creates a less complex vOLTMF instance along with a reduction in the more error-prone configuration tasks for each vOLTMF instance. This simplification is an enabler to efficiently scale vOLTMF instances within the vOLTMF service.

- Monitoring and management of the vOMCI function instances can be more effectively performed by the vOMCI service as the vOMCI function instances are scaled and maintained and ONUs are re-allocated between vOMCI functions instances.

x.1 Use of the ONU Management Proxy

The features of ONU Management Proxy defined in this Technical Report is critical to an effective deployment of the vOMCI service for the following reasons:

- ONUs are allocated by the ONU Management Proxy based on policies with associated ONU criteria not on a per ONU instance. This becomes a factor in large deployments where there are tens of millions of ONU's managed via thousands of vOMCI function instances and hundreds of instances of the vOLTMF.
- Connectivity toward the dynamic behavior of the vOMCI functions is hidden from the vOLTMF instances. This removes the need for the vOLTMF instance to know how to connect to individual vOMCI function instances (e.g., be configured with the vOMCI function instance's IP address, port, Kafka topics, gRPC credentials)
- Choices of message transport protocol (i.e., Kafka, gRPC) by vendor of a vOMCI function is hidden from the vOLTMF instances. This alleviates the need for the vOLTMF to support multiple message transport protocols as new vOMCI functions are added to support new types of ONUs or new software versions for ONUs where the vendor of the vOMCI function supports a message transport protocol that is not currently supported by the vOLTMF instances.

Appendix XI. Introductory Examples for deploying vOLTMF and vOMCI

- Description of Example:
 - TR-451 includes many options for a variety of deployment topologies and use-cases, including optional software components and protocol interfaces that may be configured in multiple directions (i.e., client-to-server optionally in either direction, or one connection in each direction concurrently). This can be confusing to implementers wishing to design or experiment with a new vOMCI deployment.
 - This example implementation is intended to simplify the range of options into an introduction to the critical components and configuration while eliminating many of the optional or complex variations.
 - This simplified topology is viable for lab or production use-cases, however implementers should read all of TR-451 after working with this example, to ensure the full range of options is understood.
 - The primary components used in this example:
 - vOMCI Function
 - vOLTMF
 - The optional components not used in this example:
 - vOMCI Proxy
 - ONU Management Function
 - Protobuf versions used:
 - There are now multiple versions of the vOMCI protobuf specifications available in the BBF repository. This example focuses on the original versions of each protobuf specification at the time of TR-451 Issue 1. (i.e., v1)
 - Initial interface provisioning for vOMCI function (and vOMCI proxy) vOLTMF-vOMCI interface:
 - In many BBF specifications for virtualized functions, an Minf NETCONF-YANG interface is used to provide all configuration and operational data access, including initial provisioning (aka “bootstrapping a VNF” as with CallHome).
 - In the case of the vOMCI Function and vOMCI Proxy, the vOLTMF-vOMCI interface must be created (including IP, port, transport details) when its container or VM is initially created. This can be accomplished as part of a bootstrap config, file, injected environment, etc. If this is created as part of a dedicated Minf, then that Minf is spun down/disabled after this first interface is online. In all cases, all configuration and operation of the vOMCI Function and/or vOMCI Proxy is performed over this vOLTMF-vOMCI interface as soon as it is online.
 - This same example can be used for multiple vendors' vOMCI instances, simply replicate the vOMCI Function-relevant steps for each subsequent different vOMCI Function instance. Multiple pOLTs can likewise connect to these multiple vOMCI instances using the same syntax in the OLT section of these configs.

- Detailed common XML examples:
- Step 1: On the vOLTMF, establish a base configuration

```
<voltmf xmlns="urn:bbf:yang:bbf-voltmf">
  <nf-topology>
    <type>point-to-point</type>
    <!--The vnf entries are like a catalog of types/flavors, not the
      actual VNF instances. There could be three such "flavors" of
      the same vendor's vOMCI-proxy, for example, if they were
```

```

    rolling out upgrades and testing a beta version. -->
<vnf>
  <name>VOLTMF_FLAVOR1</name>
  <vendor>VOLTMF_FLAVOR1_VENDOR</vendor>
  <software-version>VOLTMF_FLAVOR1_VERSION1</software-version>
  <nf-type>voltmf</nf-type>
</vnf>
<vnf>
  <name>VOMCI_FLAVOR2</name>
  <vendor>VOMCI_FLAVOR2_VENDOR</vendor>
  <software-version>VOMCI_FLAVOR1_VERSION</software-version>
  <nf-type>vomci-function</nf-type>
</vnf>
<vnf>
  <name>VOMICIPROXY_FLAVOR1</name>
  <vendor>VOMICIPROXY_FLAVOR1_VENDOR</vendor>
  <software-version>VOMICIPROXY_FLAVOR1_VERSION</software-version>
  <nf-type>vomci-proxy</nf-type>
</vnf>
<!--The vnf instances are specific, unique, named instances of
      VNFs that exist (or are perhaps up, down, etc.) If any
      instance has particular YANG schemas that are to be
      managed through the vOLTMF's YANG-over-protobuf protocol,
      then they are loaded via the "data" yang-schema-mount. -->
<vnf-instance>
  <name>VOLTMF1_NAME</name>
  <vnf>VOLTMF_FLAVOR1</vnf>
</vnf-instance>
<vnf-instance>
  <name>VOMCI1_NAME</name>
  <vnf>VOMCI_FLAVOR1</vnf>
  <data/> <!-- yang-schema-mount -- this is where the root vomci-function yang is configured -->
</vnf-instance>
<!-- The second example is not necessary for this Test Option:
<vnf-instance>
  <name>VOMCI2_NAME</name>
  <vnf>VOMCI_FLAVOR1</vnf>
  <data/> <!-- yang-schema-mount -->
</vnf-instance>
-->
<!-- The example proxy is not necessary for this Test Option:
<vnf-instance>
  <name>VOMICIPROXY1_NAME</name>
  <vnf>VOMICIPROXY_FLAVOR1</vnf>
  <data/> <!-- yang-schema-mount -->
</vnf-instance>
-->
<!--The list of nf-links are defined links between two discrete
      instances on the vnf-instance list. This list of links is
      used when an ONU under vOMCI management specifies its
      onu-management-chain, which is a list of named VNF "hops"
      used for managing it. Note that the nf-instance and endpoint
      names are specific, exactly-named instances, despite these
      not being leafrefs. i.e., it is possible to misconfigure
      an ONU to use a series of VNFs whose network endpoints /
      transport have not yet been actually configured. Order of
      operations is not explicitly important though. -->
<nf-link>
  <name>VOLTMF1-VOMCI1_KAFKA</name>
  <endpoint>
    <nf-instance>VOLTMF1_NAME</nf-instance>
    <endpoint>VOLTMF1_KAFKA1</endpoint>
    <nf-type>voltmf</nf-type>
    <nf-instance>VOMCI1_NAME</nf-instance>
    <endpoint>VOMCI1_KAFKA1</endpoint>
    <nf-type>vomci-function</nf-type>
  </endpoint>
</nf-link>
<nf-link>

```

```

    <name>VOMCI1-POLT1_GRP1</name>
    <endpoint>
      <nf-instance>VOMCI1_NAME</nf-instance>
      <endpoint>VOMCI1_GRP1</endpoint>
      <nf-type>vomci-function</nf-type>
      <nf-instance>OLT1_NAME</nf-instance>
      <endpoint>OLT1_GRP1</endpoint>
      <nf-type>olt</nf-type>
    </endpoint>
  </nf-link>
</nf-topology>
<onu-vomci-assignment-policy>
  <rule>
    <!-- list - 1 to n, 1 is highest priority -->
    <name>RULE10_MATCH_ANY</name>
    <priority>10</priority>
    <match-criteria>
      <!-- onu-endpoint-criteria = any-onu, onu-vendor, and/or onu-software-version -->
      <criteria-type>bbf-vomci:any-onu</criteria-type>
      <!-- <onu-vendor> -->
      <!-- type: bbf-vomci:onu-vendor-id -
            4-character vendor ID of the TC layer ONU S/N -->
      <!-- </onu-vendor> -->
      <!-- <onu-software-version> -->
      <!-- type: bbf-vomci:onu-software-version -
            string-ascii64 -->
      <!-- </onu-software-version> -->
    </match-criteria>
    <!-- vomci-function here is actually a leaf-list of one or
          more "FLAVORS" that an ONU would get steered to, if
          there was not already a configured path for it. -->
    <vomci-function>VOMCI_FLAVOR1</vomci-function>
  </rule>
</onu-vomci-assignment-policy>
</volumf>

<!-- Note that the device-aggregation configuration is not used until pOLT,
      ONU, and other records are needed. Shown here for reference, at the
      same root level as the VOLUMF:
<devices xmlns="urn:bbf:yang:bbf-device-aggregation">
  <device/>
  <device/>
  <device/>
  <device/>
</devices>
-->

```

- Step 2: On the VOLUMF, create the vOMCI endpoint config (choosing a Kafka VOLUMF-vOMCI interface):

```

<volumf xmlns="urn:bbf:yang:bbf-volumf">
  <remote-nf>
    <!-- This is the VOLUMF-vOMCI interface, either gRPC or kafka -->
    <nf-client>
      <enabled>true</enabled>
      <initiate>
        <remote-server>
          <name>VOMCI1-VOLUMF1_KAFKA1</name>
          <nf-type>vomci-function</nf-type>
          <on-demand>false</on-demand>
          <local-service-endpoint>VOLUMF1_KAFKA1</local-service-endpoint>
          <kafka-agent xmlns="urn:bbf:yang:bbf-volumf-kafka-agent">
            <client-id>VOLUMF1_NAME</client-id>
            <publication-parameters>
              <topic>
                <name>VOMCI1_volumf-vomci-request</name>
                <!--Note: despite being all-caps, this is a pre-
                     defined value in bbf-kafka-agent.yang. The
                     values are: VOMCI_NOTIFICATION, VOMCI_REQUEST,
                     VOMCI_RESPONSE -->
              </topic>
            </publication-parameters>
          </kafka-agent>
        </remote-server>
      </initiate>
    </nf-client>
  </remote-nf>
</volumf>

```

```

        <purpose>VOMCI_REQUEST</purpose>
        <!-- partition is optional, and not very useful at
            low volumes. -->
        <partition>PARTITION1</partition>
    </topic>
</publication-parameters>
<consumption-parameters>
    <!-- Reminder that if for some reason you wanted
        multiple consumers to read the same partition
        data from the same topic, you'd use different
        group-ids. In our case, the topics will always
        be unique because we're using topics for a kind
        of point-to-point communications. -->
    <group-id>GROUP1</group-id>
    <topic>
        <name>VOMCI1_voltmf-vomci-response</name>
        <purpose>VOMCI_RESPONSE</purpose>
        <partition>PARTITION1</partition>
    </topic>
    <topic>
        <name>VOMCI1_voltmf-vomci-notification</name>
        <purpose>VOMCI_NOTIFICATION</purpose>
        <partition>PARTITION1</partition>
    </topic>
</consumption-parameters>
<access-point>
    <!-- Defining more than one results in HA. -->
    <name>ZOOKEEPER1</name>
    <kafka-agent-transport-parameters>
        <tcp-client-parameters xmlns="urn:bbf:yang:bbf-voltmf-kafka-agent-tcp">
            <remote-address>ZOOKEEPER1_IPADDR</remote-address>
            <remote-port>9092</remote-port>
            <keepalives>
                <idle-time>300</idle-time>
                <max-probes>3</max-probes>
                <probe-interval>10</probe-interval>
            </keepalives>
        </tcp-client-parameters>
    </kafka-agent-transport-parameters>
</access-point>
</kafka-agent>
</remote-server>
</initiate>
</nf-client>
</remote-nf>
</voltmf>

```

- Step 3: The vOMCI-Function needs to be bootstrapped with sufficient connectivity info that it is able to connect to the vOLTMF over the vOLTMF-vOMCI interface. This XML example shows the attributes of this base config, but this is not the only way to initiate that. The vOLTMF may replace another bootstrap config with this in a YANG datastore, for example.

```

<voltmf xmlns="urn:bbf:yang:bbf-voltmf">
    <nf-topology>
        <vnf-instance>
            <name>VOMCI1_NAME</name>
            <data>
                <vomci xmlns="urn:bbf:yang:bbf-vomci-function">
                    <message-timeout>900</message-timeout>
                    <omci-message-retransmission>
                        <enabled>true</enabled>
                        <low-priority-transmission-timeout>1000</low-priority-transmission-timeout>
                        <high-priority-transmission-timeout>1000</high-priority-transmission-timeout>
                        <low-priority-message-retries>3</low-priority-message-retries>
                        <high-priority-message-retries>3</high-priority-message-retries>
                    </omci-message-retransmission>
                    <remote-nf>
                        <nf-client>

```

```

    <enabled>true</enabled>
    <initiate>
      <remote-server>
        <name>VOLTMF1-VOMCI1_KAFKA1</name>
        <nf-type>voltmf</nf-type>
        <on-demand>false</on-demand>
        <local-service-endpoint>VOMCI1_KAFKA1</local-service-endpoint>
        <kafka-agent xmlns="urn:bbf:yang:bbf-vomci-function-kafka-agent">
          <client-id>VOMCI1_NAME</client-id>
          <publication-parameters>
            <topic>
              <name>VOMCI1_voltmf-vomci-response</name>
              <purpose>VOMCI_RESPONSE</purpose>
              <partition>PARTITION1</partition>
            </topic>
            <topic>
              <name>VOMCI1_voltmf-vomci-notification</name>
              <purpose>VOMCI_NOTIFICATION</purpose>
              <partition>PARTITION1</partition>
            </topic>
          </publication-parameters>
          <consumption-parameters>
            <group-id>GROUP1</group-id>
            <topic>
              <name>VOMCI1_voltmf-vomci-request</name>
              <purpose>VOMCI_REQUEST</purpose>
              <partition>PARTITION1</partition>
            </topic>
          </consumption-parameters>
          <access-point>
            <name>ZOOKEEPER1</name>
            <kafka-agent-transport-parameters>
              <tcp-client-parameters xmlns="urn:bbf:yang:bbf-vomci-function-kafka-agent-tcp">
                <remote-address>ZOOKEEPER1_IPADDR</remote-address>
                <remote-port>9092</remote-port>
                <keepalives>
                  <idle-time>300</idle-time>
                  <max-probes>3</max-probes>
                  <probe-interval>10</probe-interval>
                </keepalives>
              </tcp-client-parameters>
            </kafka-agent-transport-parameters>
          </access-point>
        </kafka-agent>
      </remote-server>
    </initiate>
  </nf-client>
</remote-nf>
</vomci>
</data>
</vnf-instance>
</nf-topology>
</voltmf xmlns="urn:bbf:yang:bbf-voltmf">

```

- On the VOLTMF, configure the vOMCI-Function's endpoint for the POLT's connection:

```

<voltmf xmlns="urn:bbf:yang:bbf-voltmf">
  <nf-topology>
    <vnf-instance>
      <name>VOMCI1_NAME</name>
      <data>
        <vomci xmlns="urn:bbf:yang:bbf-vomci-proxy">
          <remote-nf>
            <nf-server>
              <enabled>true</enabled>
              <listen>
                <idle-timeout>3600</idle-timeout>
                <listen-endpoint>
                  <name>VOMCI1-POLT1_GRPCSERVER1</name>

```

```

    <local-service-endpoint>VOMCI1_GRPCSERVER1</local-service-endpoint>
    <grpc-server xmlns="urn:bbf:yang:bbf-vomci-function-grpc-server">
      <tcp-server-parameters xmlns="urn:bbf:yang:bbf-vomci-function-grpc-server-tcp">
        <local-address>VOMCI1_GRPCSERVER1_IPADDR</local-address>
        <local-port>8100</local-port>
        <keepalives>
          <idle-time>300</idle-time>
          <max-probes>3</max-probes>
          <probe-interval>10</probe-interval>
        </keepalives>
      </tcp-server-parameters>
    </grpc-server>
  </listen-endpoint>
</listen>
</nf-server>
</remote-nf>
</vomci>
</data>
</vnf-instance>
</nf-topology>
</voltmf xmlns="urn:bbf:yang:bbf-voltmf">

```

- On the VOLTMF, create the NETCONF-YANG Minf mount point for managing the pOLT:

```

<devices xmlns="urn:bbf:yang:bbf-device-aggregation">
  <device>
    <name>OLT1_NAME</name>
    <type>olt</type>
    <identification>
      <mfg-name>OLT1_MANUFACTURER</mfg-name>
      <model-name>OLT1_MODEL</model-name>
      <serial-num>OLT1_SERIAL</serial-num>
    </identification>
    <remote-nf xmlns="urn:bbf:yang:bbf-device-aggregation-connectivity">
      <nf-client>
        <enabled>true</enabled>
        <initiate>
          <remote-server>
            <name>POLT1_SSH</name>
            <nf-type>olt</nf-type>
            <on-demand>false</on-demand>
            <local-service-endpoint>VOLTMF1_SSH1</local-service-endpoint>
            <netconf-client xmlns="urn:bbf:yang:bbf-device-aggregation-connectivity-netconf-client">
              <access-point>
                <name>POLT1_SSH1</name>
                <netconf-transport-parameters>
                  <initiate>
                    <netconf-server>
                      <name>POLT1_SSH1</name>
                      <endpoints>
                        <endpoint>
                          <name>POLT1_SSH1</name>
                          <ssh>
                            <tcp-client-parameters>
                              <remote-address>POLT1_SSH1_IPADDR</remote-address>
                              <remote-port>830</remote-port>
                              <keepalives>
                                <idle-time>300</idle-time>
                                <max-probes>3</max-probes>
                                <probe-interval>10</probe-interval>
                              </keepalives>
                            </tcp-client-parameters>
                            <ssh-client-parameters>
                              <client-identity>
                                <username>POLT1_SSH1_USERNAME</username>
                                <password>
                                  <cleartext-password>POLT1_SSH1_PASSWORD</cleartext-password>
                                </password>
                              </client-identity>

```

```

        <keepalives>
          <max-wait>30</max-wait>
          <max-attempts>3</max-attempts>
        </keepalives>
      </ssh-client-parameters>
    </netconf-client-parameters/>
  </ssh>
</endpoint>
</endpoints>
<connection-type>
  <persistent/>
</connection-type>
<reconnect-strategy>
  <start-with>first-listed</start-with>
  <max-wait>5</max-wait>
  <max-attempts>3</max-attempts>
</reconnect-strategy>
</netconf-server>
</initiate>
</netconf-transport-parameters>
</access-point>
</netconf-client>
</remote-server>
</initiate>
</nf-client>
</remote-nf>
</device>
</devices>

```

- On the vOLTMF, edit the pOLT's managed config, adding in the endpoint for the vOMCI-Function connectivity:

```

<devices xmlns="urn:bbf:yang:bbf-device-aggregation">
  <device>
    <name>OLT1_NAME</name>
    <data>
      <remote-nf xmlns="urn:bbf:yang:bbf-olt-vomci">
        <nf-client>
          <enabled>true</enabled>
          <initiate>
            <remote-server>
              <name>POLT1-VOMCI1_GRPCLIENT1</name>
              <nf-type>vomci-function</nf-type>
              <on-demand>false</on-demand>
              <local-service-endpoint>POLT1_GRPCLIENT1</local-service-endpoint>
              <grpc-client xmlns="urn:bbf:yang:bbf-olt-vomci-grpc-client">
                <channel>
                  <ping-interval>300</ping-interval>
                </channel>
                <connection-backoff>
                  <enable>true</enable>
                  <initial-backoff>30</initial-backoff>
                  <min-connect-timeout>20</min-connect-timeout>
                  <multiplier>1.60</multiplier>
                  <jitter>0.20</jitter>
                  <max-backoff>10</max-backoff>
                </connection-backoff>
                <access-point>
                  <name>VOMCI1_GRPCTSERVER1</name>
                  <grpc-transport-parameters>
                    <tcp-client-parameters xmlns="urn:bbf:yang:bbf-olt-vomci-grpc-client-tcp">
                      <remote-address>VOMCI1_GRPCTSERVER1_IPADDR</remote-address>
                      <remote-port>8100</remote-port>
                      <keepalives>
                        <idle-time>300</idle-time>
                        <max-probes>3</max-probes>
                        <probe-interval>10</probe-interval>
                      </keepalives>
                    </tcp-client-parameters>
                  </grpc-transport-parameters>

```



```
        </access-point>
      </grpc-client>
    </remote-server>
  </initiate>
</nf-client>
<nf-endpoint-filter>
  <rule>
    <name>POLT1_RULE10_match-any-onu</name>
    <priority>10</priority>
    <match-criteria>
      <criteria-type>bbf-vomcit:any-onu</criteria-type>
      <!--<onu-vendor>-->
        <!-- type: bbf-vomcit:onu-vendor-id -->
        <!--</onu-vendor>-->
      </match-criteria>
      <endpoint>VOMCI_FLAVOR1</endpoint>
    </rule>
  </nf-endpoint-filter>
</remote-nf>
</data>
</device>
</devices>
```

End of Broadband Forum Technical Report TR-451