



ISO/IEC 24775-3

Edition 1.0 2014-10

INTERNATIONAL STANDARD



Information technology – Storage management –
Part 3: Common profiles

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2014 ISO/IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester. If you have any questions about ISO/IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

IEC Central Office
3, rue de Varembe
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00
info@iec.ch
www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

IEC Catalogue - webstore.iec.ch/catalogue

The stand-alone application for consulting the entire bibliographical information on IEC International Standards, Technical Specifications, Technical Reports and other documents. Available for PC, Mac OS, Android Tablets and iPad.

IEC publications search - www.iec.ch/searchpub

The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee,...). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished

Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and also once a month by email.

Electropedia - www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 30 000 terms and definitions in English and French, with equivalent terms in 14 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary

More than 55 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

IEC Customer Service Centre - webstore.iec.ch/csc

If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: csc@iec.ch.

IECNORM.COM : Click to view the full text of ISO/IEC 24775-3:2014



ISO/IEC 24775-3

Edition 1.0 2014-10

INTERNATIONAL STANDARD



Information technology – Storage management –
Part 3: Common profiles

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

PRICE CODE

H

ICS 35.200

ISBN 978-2-8322-1894-5

Warning! Make sure that you obtained this publication from an authorized distributor.

CONTENTS

FOREWORD.....	35
Introduction	37
1 Scope	39
2 Normative references	39
3 Terms, definitions and abbreviations.....	41
3.1 Terms and definitions.....	41
3.2 Abbreviations	41
4 Profile introduction.....	42
4.1 Profile overview.....	42
4.1.1 General.....	42
4.1.2 Terminology.....	43
4.2 Format for Profile Specifications	43
5 Typographical Conventions.....	46
5.1 Maturity Model.....	46
5.2 Experimental Maturity Level.....	46
5.3 Implemented Maturity Level.....	46
5.4 Stable Maturity Level.....	47
5.5 Finalized Maturity Level	47
5.6 Deprecated Material.....	47
6 Recipe overview	49
6.1 Recipe concepts.....	49
6.2 Recipe Pseudo Code conventions	49
6.2.1 Overview	49
6.2.2 General Syntax.....	49
6.2.3 CIM related variable and methods	50
6.2.4 Data structure.....	51
6.2.5 Operations.....	51
6.2.6 Control operations.....	52
6.2.7 Functions.....	52
6.2.8 Exception handling	53
6.2.9 Built-in functions	53
6.2.10 Extrinsic method calls.....	54
7 Generic Target Ports Profile.....	55
7.1 Synopsis.....	55
7.2 Description	55
7.3 Implementation.....	55
7.3.1 General.....	55
7.3.2 Modeling SCSI/SB Logical Units	57
7.4 Methods of the Profile	57
7.4.1 Extrinsic methods	57
7.4.2 Intrinsic methods	58
7.5 Use Cases.....	58
7.6 CIM Elements.....	58
7.6.1 General.....	58
7.6.2 CIM_DeviceSAPImplementation	58
7.6.3 CIM_HostedAccessPoint.....	59
7.6.4 CIM_LogicalPort.....	59
7.6.5 CIM_ProtocolEndpoint	59
7.6.6 CIM_SystemDevice (Port).....	60
8 Parallel SCSI (SPI) Target Ports Profile	61

8.1	Synopsis.....	61
8.2	Description	61
8.3	Implementation.....	61
8.4	Health and Fault Management.....	62
8.5	Methods – Extrinsic Methods of this Subprofile	62
8.6	CIM elements	62
8.6.1	General.....	62
8.6.2	CIM_DeviceSAPImplementation	62
8.6.3	CIM_HostedAccessPoint.....	63
8.6.4	CIM_SCSIProtocolEndpoint	63
8.6.5	CIM_SPIPort	64
8.6.6	CIM_SystemDevice (Port).....	64
9	FC Target Ports Profile.....	66
9.1	Synopsis.....	66
9.2	Description	66
9.3	Implementation.....	66
9.3.1	General.....	66
9.3.2	SMI-S 1.0 backwards compatibility	66
9.4	Durable Names and Correlatable IDs of the Subprofile	67
9.5	Health and Fault Management.....	67
9.6	Supported Profiles and Packages.....	67
9.7	Extrinsic Methods of this Subprofile	67
9.8	Client Considerations and Recipes	67
9.9	CIM Elements.....	68
9.9.1	General.....	68
9.9.2	CIM_DeviceSAPImplementation	68
9.9.3	CIM_FCPort	69
9.9.4	CIM_HostedAccessPoint.....	70
9.9.5	CIM_ProtocolControllerForPort	70
9.9.6	CIM_SCSIProtocolEndpoint	70
9.9.7	CIM_SystemDevice (Port).....	71
10	iSCSI Target Ports Subprofile.....	72
10.1	Synopsis.....	72
10.2	Description	72
10.3	Implementation.....	72
10.3.1	General.....	72
10.3.2	Mapping and Masking considerations	74
10.3.3	Settings	75
10.3.4	Durable Names and Correlatable IDs of the Subprofile	75
10.4	Health and Fault Management.....	75
10.5	Supported Subprofiles and Packages.....	76
10.6	Methods of this Subprofile.....	76
10.6.1	General.....	76
10.6.2	CreateiSCSINode	76
10.6.3	DeleteiSCSINode	77
10.6.4	CreateiSCSIProtocolEndpoint	77
10.6.5	DeleteiSCSIProtocolEndpoint	79
10.6.6	BindiSCSIProtocolEndpoint.....	79
10.7	Client Considerations and Recipes	80
10.7.1	Discover the iSCSI Target Port capabilities	80
10.7.2	Identify the iSCSI Nodes in a target system	81
10.7.3	Identify the iSCSI Ports on a given iSCSI node	81
10.7.4	Identify the iSCSI sessions existing on an iSCSI node	82
10.7.5	Create an iSCSI Target Node on an iSCSI Network Entity	83

10.7.6	Create an iSCSI Target Port on an iSCSI target node	84
10.7.7	Add a Network Portal to a Target Port	85
10.7.8	Determine the health of Nodes in a target system	87
10.7.9	Determine the health of a Session on a target system	87
10.7.10	Configure the default settings for Sessions created in a target computer system ..	88
10.7.11	Configure default settings for Connections on Network Portals used by an iSCSIProtocolEndpoint	89
10.7.12	Get the statistics for a Session on a target system	89
10.7.13	Configure Enable/disable header and data digest	89
10.8	CIM Elements	90
10.8.1	General	90
10.8.2	CIM_BindsTo (TCPProtocolEndpoint to IPProtocolEndpoint)	92
10.8.3	CIM_BindsTo (iSCSIProtocolEndpoint to TCPProtocolEndpoint)	92
10.8.4	CIM_ConcreteDependency	93
10.8.5	CIM_DeviceSAPImplementation (EthernetPort to IPProtocolEndpoint)	93
10.8.6	CIM_DeviceSAPImplementation (EthernetPort to iSCSIProtocolEndpoint)	93
10.8.7	CIM_ElementCapabilities (iSCSIConfigurationCapabilities to System)	94
10.8.8	CIM_ElementCapabilities (iSCSIConfigurationCapabilities to iSCSIConfigurationService)	94
10.8.9	CIM_ElementSettingData (iSCSIConnectionSettings to TCPProtocolEndpoint)	94
10.8.10	CIM_ElementSettingData (iSCSIConnectionSettings to iSCSIProtocolEndpoint) ..	95
10.8.11	CIM_ElementSettingData (iSCSIConnectionSettings to SCSIProtocolController)	95
10.8.12	CIM_ElementSettingData (iSCSIConnectionSettings to System)	95
10.8.13	CIM_ElementSettingData (iSCSIConnectionSettings to iSCSIProtocolEndpoint)	96
10.8.14	CIM_ElementStatisticalData (iSCSILoginStatistics to SCSIProtocolController)	96
10.8.15	CIM_ElementStatisticalData (iSCSIConnectionFailures to SCSIProtocolController) ..	96
10.8.16	CIM_ElementStatisticalData (iSCSIConnectionStatistics to iSCSIConnection)	97
10.8.17	CIM_EndpointOfNetworkPipe (iSCSIConnection to TCPProtocolEndpoint)	97
10.8.18	CIM_EndpointOfNetworkPipe (iSCSIConnection to iSCSIProtocolEndpoint)	98
10.8.19	CIM_EthernetPort	98
10.8.20	CIM_HostedAccessPoint (System to IPProtocolEndpoint)	98
10.8.21	CIM_HostedAccessPoint (System to TCPProtocolEndpoint)	99
10.8.22	CIM_HostedAccessPoint (System to iSCSIProtocolEndpoint)	99
10.8.23	CIM_HostedCollection	99
10.8.24	CIM_HostedService	100
10.8.25	CIM_IPProtocolEndpoint	100
10.8.26	CIM_MemberOfCollection	100
10.8.27	CIM_NetworkPipeComposition	101
10.8.28	CIM_SAPAvailableForElement	101
10.8.29	CIM_SCSIProtocolController	101
10.8.30	CIM_SystemDevice (System to EthernetPort)	102
10.8.31	CIM_SystemDevice (System to SCSIProtocolController)	102
10.8.32	CIM_SystemSpecificCollection	102
10.8.33	CIM_TCPProtocolEndpoint	103
10.8.34	CIM_iSCSICapabilities	103
10.8.35	CIM_iSCSIConfigurationCapabilities	104
10.8.36	CIM_iSCSIConfigurationService	104
10.8.37	CIM_iSCSIConnection	104
10.8.38	CIM_iSCSIConnectionSettings	105
10.8.39	CIM_iSCSILoginStatistics	106
10.8.40	CIM_iSCSIProtocolEndpoint	106
10.8.41	CIM_iSCSIConnection	107
10.8.42	CIM_iSCSIConnectionFailures	108
10.8.43	CIM_iSCSIConnectionSettings	108
10.8.44	CIM_iSCSIConnectionStatistics	109

11	Serial Attached SCSI (SAS) Target Port Subprofile	110
11.1	Synopsis.....	110
11.2	Description	110
11.2.1	General.....	110
11.2.2	Health and Fault Management	111
11.3	Methods	111
11.3.1	Extrinsic Methods of this Subprofile	111
11.3.2	Intrinsic Methods of this Subprofile	111
11.4	Client considerations and Recipes	111
11.5	CIM Elements.....	112
11.5.1	General.....	112
11.5.2	CIM_ConcreteComponent.....	112
11.5.3	CIM_DeviceSAPImplementation	112
11.5.4	CIM_HostedAccessPoint.....	113
11.5.5	CIM_SASPort	113
11.5.6	CIM_SCSIProtocolEndpoint	114
11.5.7	CIM_SystemDevice (Port).....	114
11.5.8	CIM_SystemDevice (SAS PHY).....	115
11.5.9	SNIA_SASPHY	115
12	Serial ATA (SATA) Target Ports Profile.....	116
12.1	Synopsis.....	116
12.2	Description	116
12.2.1	General.....	116
12.2.2	Health and Fault Management	117
12.3	Methods of this Subprofile.....	117
12.4	Client considerations and Recipes	117
12.5	CIM elements	118
12.5.1	General.....	118
12.5.2	CIM_ATAPort	118
12.5.3	CIM_ATAProtocolEndpoint	119
12.5.4	CIM_DeviceSAPImplementation	119
12.5.5	CIM_HostedAccessPoint.....	119
12.5.6	CIM_SystemDevice (Port).....	120
13	SB Target Ports Profile.....	121
13.1	Synopsis.....	121
13.2	Description	121
13.3	Implementation.....	121
13.4	Health and Fault Management Consideration.....	122
13.5	Cascading Considerations	122
13.6	Methods of the Profile	123
13.6.1	Extrinsic Methods of the Profile	123
13.6.2	Intrinsic Methods of the Profile	123
13.7	Client Considerations and Recipes	123
13.8	CIM Elements.....	123
13.8.1	General.....	123
13.8.2	CIM_DeviceSAPImplementation	124
13.8.3	CIM_FCPort	124
13.8.4	CIM_HostedAccessPoint.....	125
13.8.5	CIM_SystemDevice (Port).....	125
13.8.6	SNIA_SBProtocolEndpoint.....	125
14	Direct Attach (DA) Ports Profile	128
14.1	Description	128
14.2	Health and Fault Management.....	128
14.3	Supported Profiles and Packages.....	128

14.4	Extrinsic Methods	128
14.5	Client Considerations and Recipes	129
14.6	Registered Name and Version	129
14.7	CIM Elements.....	129
14.7.1	General.....	129
14.7.2	CIM_DAPort	129
14.7.3	CIM_DeviceSAPImplementation	130
14.7.4	CIM_HostedAccessPoint	130
14.7.5	CIM_SCSIProtocolEndpoint	130
14.7.6	CIM_SystemDevice (Port).....	131
15	Generic Initiator Ports Profile.....	132
15.1	Synopsis.....	132
15.2	Description	132
15.3	Implementation.....	132
15.3.1	General.....	132
15.3.2	Remote Device Models	133
15.3.3	Health and Fault Management Considerations	136
15.3.4	Cascading Considerations	136
15.4	Methods	136
15.4.1	Extrinsic Methods of this Profile	136
15.4.2	Intrinsic Methods of this Profile	136
15.5	Use Cases.....	136
15.6	CIM Elements.....	137
15.6.1	General.....	137
15.6.2	CIM_ConnectivityCollection	137
15.6.3	CIM_DeviceSAPImplementation	138
15.6.4	CIM_ElementStatisticalData (Port Statistics)	138
15.6.5	CIM_HostedAccessPoint (Initiator)	138
15.6.6	CIM_HostedAccessPoint (Target).....	139
15.6.7	CIM_HostedCollection (Connectivity Collection).....	139
15.6.8	CIM_LogicalPort.....	139
15.6.9	CIM_MemberOfCollection (Connectivity Collection)	140
15.6.10	CIM_ProtocolEndpoint (Initiator).....	140
15.6.11	CIM_ProtocolEndpoint (Target)	141
15.6.12	CIM_SystemDevice (Initiator Ports).....	141
15.6.13	SNIA_LogicalPortStatistics	142
16	Parallel SCSI (SPI) Initiator Ports Profile.....	143
16.1	Synopsis.....	143
16.2	Description	143
16.3	Implementation.....	143
16.3.1	General.....	143
16.3.2	Health and Fault Management Considerations	143
16.3.3	Cascading Considerations	144
16.4	Methods	144
16.4.1	Extrinsic Methods of this Profile	144
16.4.2	Intrinsic Methods of this Profile	144
16.5	Detailed Use Cases and Recipes	144
16.6	CIM Elements.....	144
16.6.1	General.....	144
16.6.2	CIM_ConnectivityCollection	145
16.6.3	CIM_DeviceSAPImplementation	145
16.6.4	CIM_ElementStatisticalData (Port Statistics)	146
16.6.5	CIM_HostedAccessPoint (Initiator)	146
16.6.6	CIM_HostedAccessPoint (Target).....	146

16.6.7	CIM_HostedCollection (Connectivity Collection).....	147
16.6.8	CIM_MemberOfCollection (Connectivity Collection)	147
16.6.9	CIM_SCSIInitiatorTargetLogicalUnitPath	148
16.6.10	CIM_SCSIProtocolEndpoint (Initiator).....	148
16.6.11	CIM_SCSIProtocolEndpoint (Target)	149
16.6.12	CIM_SPIPort	149
16.6.13	CIM_SystemDevice (Initiator Ports)	150
16.6.14	SNIA_LogicalPortStatistics	150
17	iSCSI Initiator Port Profile	152
17.1	Synopsis.....	152
17.2	Description	152
17.3	Implementation.....	152
17.3.1	General.....	152
17.3.2	Health and Fault Management Considerations	153
17.3.3	Cascading Considerations	153
17.4	Methods	153
17.4.1	Extrinsic Methods of this Profile	153
17.4.2	Intrinsic Methods of this Profile	153
17.5	Detailed Use Cases and Recipes	154
17.6	CIM Elements.....	154
17.6.1	General.....	154
17.6.2	CIM_BindsTo (Host Hardware RAID Controller).....	155
17.6.3	CIM_DeviceSAPImplementation (IPProtocolEndpoint to EthernetPort).....	155
17.6.4	CIM_DeviceSAPImplementation (iSSIPProtocolEndpoint to EthernetPort).....	155
17.6.5	CIM_EthernetPort (Host Hardware RAID Controller)	156
17.6.6	CIM_HostedAccessPoint (System to IPProtocolEndpoint)	156
17.6.7	CIM_HostedAccessPoint (System to TCPProtocolEndpoint).....	156
17.6.8	CIM_HostedAccessPoint (System to iSCSIProtocolEndpoint).....	157
17.6.9	CIM_IPProtocolEndpoint (Host Hardware RAID Controller).....	157
17.6.10	CIM_LogicalDevice (Host Hardware RAID Controller).....	158
17.6.11	CIM_SystemDevice (System to EthernetPort)	158
17.6.12	CIM_SystemDevice (System to LogicalDevice).....	158
17.6.13	CIM_TCPProtocolEndpoint (Host Hardware RAID Controller)	159
17.6.14	CIM_iSCSIProtocolEndpoint (Host Hardware RAID Controller)	159
18	FC Initiator Ports Profile	161
18.1	Synopsis.....	161
18.2	Description	161
18.3	Implementation.....	161
18.3.1	General.....	161
18.3.2	Port Statistics	162
18.3.3	Logical Port Group (FC Node).....	162
18.3.4	Health and Fault Management Considerations	163
18.3.5	Cascading Considerations	163
18.4	Methods	163
18.4.1	Extrinsic Methods of this Profile	163
18.4.2	Intrinsic Methods of this Profile	163
18.5	Use Cases – Get the statistics for each FC port	163
18.6	CIM Elements.....	164
18.6.1	General.....	164
18.6.2	CIM_ConnectivityCollection	165
18.6.3	CIM_DeviceSAPImplementation	165
18.6.4	CIM_ElementStatisticalData (Port Statistics)	166
18.6.5	CIM_FCPort	166
18.6.6	CIM_FCPortStatistics	168

18.6.7	CIM_HostedAccessPoint (Initiator)	168
18.6.8	CIM_HostedAccessPoint (Target)	169
18.6.9	CIM_HostedCollection (Connectivity Collection)	169
18.6.10	CIM_MemberOfCollection (Connectivity Collection)	170
18.6.11	CIM_ProtocolControllerForPort	170
18.6.12	CIM_SCSIInitiatorTargetLogicalUnitPath	170
18.6.13	CIM_SCSIProtocolController	171
18.6.14	CIM_SCSIProtocolEndpoint (Initiator)	171
18.6.15	CIM_SCSIProtocolEndpoint (Target)	172
18.6.16	CIM_SystemDevice (Initiator Ports)	172
19	SAS Initiator Ports Profile	174
19.1	Synopsis	174
19.2	Description	174
19.2.1	General	174
19.2.2	Health and Fault Management Considerations	175
19.3	Methods of the profile	175
19.4	Client Considerations and Recipes	175
19.5	CIM Elements	175
19.5.1	General	175
19.5.2	CIM_ATAProtocolEndpoint (Initiator)	176
19.5.3	CIM_BindsTo	176
19.5.4	CIM_ConcreteComponent	177
19.5.5	CIM_ConnectivityCollection	177
19.5.6	CIM_DeviceSAPImplementation	177
19.5.7	CIM_ElementStatisticalData (PHY Statistics)	178
19.5.8	CIM_ElementStatisticalData (Port Statistics)	178
19.5.9	CIM_HostedAccessPoint (Initiator)	178
19.5.10	CIM_HostedAccessPoint (Target)	179
19.5.11	CIM_HostedCollection (Connectivity Collection)	179
19.5.12	CIM_MemberOfCollection (Connectivity Collection)	179
19.5.13	CIM_SASPort	180
19.5.14	CIM_SCSIInitiatorTargetLogicalUnitPath	180
19.5.15	CIM_SCSIProtocolEndpoint (Initiator)	181
19.5.16	CIM_SCSIProtocolEndpoint (Target)	181
19.5.17	CIM_SystemDevice (Initiator PHY)	182
19.5.18	CIM_SystemDevice (Initiator Ports)	182
19.5.19	SNIA_LogicalPortStatistics	183
19.5.20	SNIA_SASPHY	183
19.5.21	SNIA_SASPhyStatistics	184
20	ATA Initiator Ports Profile	185
20.1	Synopsis	185
20.2	Description	185
20.3	Implementation	185
20.3.1	General	185
20.3.2	Health and Fault Management Consideration	186
20.3.3	Cascading Considerations	186
20.4	Methods of the Profile	186
20.4.1	Extrinsic Methods of the Profile	186
20.4.2	Intrinsic Methods of this Profile	186
20.5	Client Considerations and Recipes	186
20.6	CIM Elements	187
20.6.1	General	187
20.6.2	CIM_ATAInitiatorTargetLogicalUnitPath	187
20.6.3	CIM_ATAPort	188

20.6.4	CIM_ATAProtocolEndpoint (Initiator)	188
20.6.5	CIM_ATAProtocolEndpoint (Target)	189
20.6.6	CIM_ConnectivityCollection	190
20.6.7	CIM_DeviceSAPImplementation	190
20.6.8	CIM_ElementStatisticalData (Port Statistics)	190
20.6.9	CIM_HostedAccessPoint (Initiator)	191
20.6.10	CIM_HostedAccessPoint (Target)	191
20.6.11	CIM_HostedCollection (Connectivity Collection)	191
20.6.12	CIM_MemberOfCollection (Connectivity Collection)	192
20.6.13	CIM_SystemDevice (Initiator Ports)	192
20.6.14	SNIA_LogicalPortStatistics	193
21	FC-SB-x Initiator Ports Profile	194
21.1	Synopsis	194
21.2	Description	194
21.3	Implementation	194
21.3.1	General	194
21.3.2	Health and Fault Management Considerations	195
21.3.3	Cascading Considerations	195
21.4	Methods	195
21.4.1	Extrinsic Methods of the Profile	195
21.4.2	Intrinsic Methods of this Profile	195
21.5	Client Considerations and Recipes	195
21.6	CIM Elements	196
21.6.1	General	196
21.6.2	CIM_ConnectivityCollection	196
21.6.3	CIM_DeviceSAPImplementation	197
21.6.4	CIM_ElementStatisticalData (Port Statistics)	197
21.6.5	CIM_FCPort	198
21.6.6	CIM_HostedAccessPoint (Initiator)	199
21.6.7	CIM_HostedAccessPoint (Target)	199
21.6.8	CIM_HostedCollection (Connectivity Collection)	199
21.6.9	CIM_MemberOfCollection (Connectivity Collection)	200
21.6.10	CIM_SystemDevice (Initiator Ports)	200
21.6.11	SNIA_LogicalPortStatistics	200
21.6.12	SNIA_SBInitiatorTargetLogicalUnitPath	201
21.6.13	SNIA_SBPProtocolEndpoint (Initiator)	201
21.6.14	SNIA_SBPProtocolEndpoint (Target)	202
22	Backend Ports Subprofile	203
23	FCoE Initiator Ports Profile	204
23.1	Synopsis	204
23.2	Description	204
23.3	Implementation	204
23.3.1	General	204
23.3.2	Relationship to Storage HBA Profile	205
23.3.3	Optional target model	205
23.3.4	Port Statistics	205
23.3.5	Logical Port Group (FC Node)	205
23.3.6	Health and Fault Management Considerations	206
23.3.7	Cascading Considerations	206
23.4	Methods	206
23.4.1	Extrinsic Methods of this Profile	206
23.4.2	Intrinsic Methods of this Profile	206
23.5	Detailed Use Cases and Recipes	207
23.6	CIM Elements	207

23.6.1	General.....	207
23.6.2	CIM_ConnectivityCollection	208
23.6.3	CIM_DeviceSAPIImplementation	208
23.6.4	CIM_ElementStatisticalData (Port Statistics)	209
23.6.5	CIM_EthernetPort.....	209
23.6.6	CIM_FCPort	209
23.6.7	CIM_FCPortStatistics	210
23.6.8	CIM_HostedAccessPoint (Initiator)	211
23.6.9	CIM_HostedAccessPoint (Target).....	212
23.6.10	CIM_HostedCollection (Connectivity Collection).....	212
23.6.11	CIM_HostedCollection (FC Node).....	212
23.6.12	CIM_HostedDependency (NetworkPort to FCPort)	213
23.6.13	CIM_LogicalPortGroup.....	213
23.6.14	CIM_MemberOfCollection (Connectivity Collection)	213
23.6.15	CIM_MemberOfCollection (FC Node)	214
23.6.16	CIM_ProtocolEndpoint (Initiator).....	214
23.6.17	CIM_ProtocolEndpoint (Target)	215
23.6.18	CIM_SCSIInitiatorTargetLogicalUnitPath.....	215
23.6.19	CIM_SCSIProtocolEndpoint (Initiator).....	216
23.6.20	CIM_SCSIProtocolEndpoint (Target).....	216
23.6.21	CIM_SystemDevice (Ethernet Port).....	217
23.6.22	CIM_SystemDevice (Initiator Ports).....	217
24	Access Points Subprofile.....	218
24.1	Description	218
24.2	Health and Fault Management Considerations.....	219
24.3	Cascading Considerations	219
24.4	Supported Subprofiles and Packages.....	219
24.5	Methods of this Profile.....	219
24.6	Client Considerations and Recipes.....	220
24.7	Registered Name and Version	220
24.8	CIM Elements.....	220
24.8.1	General.....	220
24.8.2	CIM_HostedAccessPoint.....	220
24.8.3	CIM_RemoteServiceAccessPoint	220
24.8.4	CIM_SAPAvailableForElement	221
25	Cascading Subprofile	222
25.1	Introduction to cascading subprofile.....	222
25.2	Description	222
25.2.1	Overview	222
25.2.2	Instance Diagrams	223
25.3	Health and Fault Management Considerations.....	229
25.3.1	Reporting Health of Leaf Systems, Resources and Object Managers	229
25.3.2	Cascading Indications of Health.....	230
25.4	Cascading Considerations	230
25.5	Supported Subprofiles and Packages.....	230
25.6	Methods of this Subprofile.....	230
25.6.1	General.....	230
25.6.2	Allocate.....	230
25.6.3	Deallocate	231
25.7	Client Considerations and Recipes	232
25.7.1	Recipe MPCP01: Determining Resources used by cascading Profiles.....	232
25.7.2	Recipe MPCP02: Monitoring the existence of Cascading Profiles	232
25.7.3	OPTIONAL: Recipe MPCP03: Allocation of Leaf Resources	232
25.7.4	OPTIONAL: Recipe MPCP04: Deallocation of Leaf Resources.....	232

25.7.5	Recipe MPCP05: Monitoring the existence of “Stitching” between Profiles	232
25.7.6	Supported SNIA_CascadingCapabilities Patterns	232
25.8	Registered Name and Version	232
25.9	CIM Elements.....	233
25.9.1	General.....	233
25.9.2	CIM_ComputerSystem (Leaf System).....	235
25.9.3	CIM_Dependency (Object Managers).....	235
25.9.4	CIM_Dependency (Profile to Object Manager)	236
25.9.5	CIM_Dependency (Systems)	236
25.9.6	CIM_ElementCapabilities	236
25.9.7	CIM_ElementConformsToProfile (Leaf)	237
25.9.8	CIM_HostedCollection (Allocated Resources)	237
25.9.9	CIM_HostedCollection (Remote Resources)	238
25.9.10	CIM_HostedService (Allocation Service)	238
25.9.11	CIM_HostedService (Object Manager)	239
25.9.12	CIM_LogicalDisk	239
25.9.13	CIM_LogicalIdentity (General)	240
25.9.14	CIM_LogicalIdentity (LogicalDisk).....	241
25.9.15	CIM_LogicalIdentity (StorageVolume)	241
25.9.16	CIM_MemberOfCollection (Allocated Resources)	241
25.9.17	CIM_MemberOfCollection (Remote Resources).....	242
25.9.18	CIM_Namespace (Leaf).....	242
25.9.19	CIM_NamespaceInManager (Leaf).....	243
25.9.20	CIM_ObjectManager (Leaf).....	243
25.9.21	CIM_RegisteredProfile (Leaf).....	243
25.9.22	CIM_RemoteServiceAccessPoint (Leaf).....	244
25.9.23	CIM_SAPAvailableForElement	244
25.9.24	CIM_StorageVolume.....	245
25.9.25	CIM_SystemDevice (Leaf Devices)	246
25.9.26	SNIA_AllocatedResources.....	246
25.9.27	SNIA_AllocationService	247
25.9.28	SNIA_CascadingCapabilities	248
25.9.29	SNIA_RemoteResources	248
26	Health Package	250
26.1	Description	250
26.1.1	General.....	250
26.1.2	Error Reporting Mechanism	250
26.1.3	Event Reporting Mechanism	251
26.1.4	Standard Events.....	251
26.1.5	Reporting Health	252
26.1.6	Computer System Operational Status.....	252
26.1.7	Event Reporting.....	253
26.1.8	Fault Region	253
26.1.9	RelatedElementCausingError.....	253
26.1.10	HealthState	254
26.2	Health and Fault Management Considerations.....	254
26.3	Cascading Considerations	254
26.4	Supported Subprofiles and Packages.....	254
26.5	Client Considerations and Recipes	254
26.6	Registered Name and Version	254
26.7	CIM Elements.....	254
26.7.1	General.....	254
26.7.2	CIM_ComputerSystem	255
26.7.3	CIM_LogicalDevice	255

26.7.4	CIM_RelatedElementCausingError	256
27	Job Control Subprofile	257
27.1	Description	257
27.1.1	General.....	257
27.1.2	Instance Diagram	257
27.1.3	MethodResult	258
27.1.4	OperationalStatus for Jobs	259
27.1.5	JobState for Jobs	259
27.1.6	Determining How Long a Job Remains after Execution	260
27.2	Health and Fault Management.....	260
27.3	Cascading Considerations	260
27.4	Support Subprofiles and Packages	260
27.5	Methods of the Profile	261
27.5.1	Job Modification	261
27.5.2	Getting Error Conditions from Jobs	261
27.5.3	Suspending, Killing or Terminating a Job	261
27.6	Client Considerations and Recipes	262
27.7	Registered Name and Version	263
27.8	CIM Elements.....	263
27.8.1	General.....	263
27.8.2	CIM_AffectedJobElement.....	264
27.8.3	CIM_AssociatedJobMethodResult	264
27.8.4	CIM_ConcreteJob	264
27.8.5	CIM_MethodResult.....	266
27.8.6	CIM_OwningJobElement.....	267
28	Location Subprofile.....	268
28.1	Description	268
28.1.1	General.....	268
28.1.2	Instance Diagram	268
28.2	Health and Fault Management Considerations	268
28.3	Cascading Considerations	268
28.4	Supported Subprofiles and Packages	268
28.5	Methods of the Profile	268
28.6	Client Considerations and Recipes	268
28.7	Registered Name and Version	268
28.8	CIM Elements.....	269
28.8.1	General.....	269
28.8.2	CIM_Location	269
28.8.3	CIM_PhysicalElementLocation.....	269
29	Extra Capacity Set Subprofile.....	270
30	Cluster Subprofile	271
31	Multiple Computer System Subprofile	272
31.1	Description	272
31.1.1	General.....	272
31.1.2	Top Level System.....	272
31.1.3	Non-Top-Level Systems	273
31.1.4	Types of RedundancySets	273
31.1.5	Multiple Tiers of Systems	273
31.1.6	Associations between ComputerSystems and other Logical Elements.....	274
31.1.7	Associations between ComputerSystems and PhysicalPackages and Products..	275
31.1.8	Storage Systems without Multiple Systems	276
31.1.9	Durable Names and Correlatable IDs of the Subprofile	276
31.2	Health and Fault Management Considerations	276
31.3	Cascading Considerations	276

31.4	Supported Subprofiles and Packages	276
31.5	Methods of the Profile	276
31.6	Client Considerations and Recipes	276
31.6.1	General	276
31.6.2	Find Top-level Computer Systems	276
31.6.3	Find the Top-level Computer System for any LogicalDevice	276
31.7	Registered Name and Version	279
31.8	CIM Elements	279
31.8.1	General	279
31.8.2	CIM_ComponentCS	279
31.8.3	CIM_ComputerSystem (Non-Top-Level System)	280
31.8.4	CIM_ConcreteIdentity	280
31.8.5	CIM_IsSpare	280
31.8.6	CIM_MemberOfCollection	281
31.8.7	CIM_RedundancySet	281
32	Physical Package Package	282
32.1	Description	282
32.1.1	General	282
32.1.2	Well Defined Subcomponents	282
32.1.3	Multiple Product Identities	283
32.2	Health and Fault Management Considerations	284
32.3	Cascading Considerations	284
32.4	Supported Subprofiles and Packages	284
32.5	Methods of this Profile	284
32.6	Client Considerations and Recipes	284
32.6.1	Find Asset Information	284
32.6.2	Finding Product information	284
32.6.3	Finding Asset information	285
32.7	Registered Name and Version	285
32.8	CIM Elements	285
32.8.1	General	285
32.8.2	CIM_Container	285
32.8.3	CIM_LogicalIdentity	286
32.8.4	CIM_PhysicalElementLocation	286
32.8.5	CIM_PhysicalPackage (Component)	287
32.8.6	CIM_PhysicalPackage (System)	287
32.8.7	CIM_Product (Component)	288
32.8.8	CIM_Product (System)	288
32.8.9	CIM_ProductParentChild	289
32.8.10	CIM_ProductPhysicalComponent (Component)	289
32.8.11	CIM_ProductPhysicalComponent (System)	289
32.8.12	CIM_SystemPackaging (Component)	290
32.8.13	CIM_SystemPackaging (System)	290
33	Power Supply Profile	291
33.1	Synopsis	291
33.2	Description	291
33.3	Implementation	291
33.3.1	General	291
33.3.2	Health and Fault Management Consideration	291
33.3.3	Cascading Considerations	291
33.4	Methods	291
33.5	Use Cases	291
33.6	CIM Elements	292
33.6.1	General	292

33.6.2	CIM_ElementCapabilities	292
33.6.3	CIM_IsSpare	293
33.6.4	CIM_MemberOfCollection	294
33.6.5	CIM_OwningCollectionElement	294
33.6.6	CIM_PowerSupply	294
33.6.7	CIM_RedundancySet	295
33.6.8	CIM_SuppliesPower	296
33.6.9	CIM_SystemDevice	296
34	Fan Profile	297
34.1	Synopsis	297
34.2	Description	297
34.3	Implementation	297
34.3.1	Health and Fault Management Consideration	297
34.3.2	Cascading Considerations	297
34.4	Methods	297
34.5	Use Cases	297
34.6	CIM Elements	298
34.6.1	General	298
34.6.2	CIM_AssociatedCooling	298
34.6.3	CIM_AssociatedSensor	299
34.6.4	CIM_ElementCapabilities	299
34.6.5	CIM_EnabledLogicalElementCapabilities	299
34.6.6	CIM_Fan	300
34.6.7	CIM_IsSpare	301
34.6.8	CIM_MemberOfCollection	301
34.6.9	CIM_NumericSensor	301
34.6.10	CIM_OwningCollectionElement	302
34.6.11	CIM_RedundancySet	302
34.6.12	CIM_Sensor	303
34.6.13	CIM_SystemDevice	303
35	Sensors Profile	304
35.1	Synopsis	304
35.2	Description	304
35.3	Implementation	304
35.3.1	General	304
35.3.2	Health and Fault Management Consideration	304
35.3.3	Cascading Considerations	304
35.4	Methods	304
35.5	Use Cases	304
35.6	CIM Elements	304
35.6.1	General	304
35.6.2	CIM_AssociatedSensor	305
35.6.3	CIM_ElementCapabilities	305
35.6.4	CIM_EnabledLogicalElementCapabilities	306
35.6.5	CIM_NumericSensor	306
35.6.6	CIM_Sensor	307
35.6.7	CIM_SystemDevice	308
36	Base Server Profile	310
36.1	Synopsis	310
36.2	Description	310
36.3	Implementation	310
36.3.1	General	310
36.3.2	HBA Instrumentation	311
36.3.3	Host Hardware RAID Instrumentation	311

36.3.4	Storage Enclosure Instrumentation	311
36.3.5	Health and Fault Management Consideration	311
36.3.6	Cascading Considerations	311
36.4	Methods	311
36.5	Use Cases	311
36.6	CIM Elements	311
36.6.1	General	311
36.6.2	CIM_ComputerSystem	312
36.6.3	CIM_ComputerSystemPackage	312
36.6.4	CIM_ElementCapabilities	313
36.6.5	CIM_EnabledLogicalElementCapabilities	313
36.6.6	CIM_HostedService	313
36.6.7	CIM_PhysicalPackage	314
36.6.8	CIM_ServiceAffectsElement	315
36.6.9	CIM_TimeService	315
37	Media Access Device Profile	316
37.1	Synopsis	316
37.2	Description	316
37.2.1	General	316
37.2.2	Location Indicator	316
37.2.3	Media Access Device Online/Offline	316
37.3	Implementation	317
37.3.1	Health and Fault Management Consideration	317
37.3.2	Cascading Considerations	317
37.3.3	Hot swap insertion or Removal of Drives	318
37.4	Methods	318
37.4.1	Request State Change	318
37.5	Use Cases	318
37.6	CIM Elements	318
37.6.1	General	318
37.6.2	CIM_EnabledLogicalElementCapabilities	319
37.6.3	CIM_HostedAccessPoint	319
37.6.4	CIM_MediaAccessDevice	319
37.6.5	CIM_PhysicalPackage	320
37.6.6	CIM_ProtocolEndpoint	320
37.6.7	CIM_Realizes	321
37.6.8	CIM_SAPAvailableForElement	321
37.6.9	CIM_SystemDevice	321
38	Storage Enclosure Profile	323
38.1	Synopsis	323
38.2	Description	323
38.2.1	General	323
38.2.2	Guidelines related to Referencing Profiles	323
38.2.3	Examples of Storage Enclosure Configurations	325
38.3	Implementation	326
38.3.1	Health and Fault Management Consideration	326
38.3.2	Cascading Considerations	326
38.3.3	Enclosure Elements	326
38.3.4	Storage Elements	326
38.3.5	Physical Assets	327
38.4	Methods	328
38.4.1	Extrinsic Methods of the Profile	328
38.4.2	Intrinsic Methods of this Profile	329
38.5	Use Cases	329

38.6	CIM Elements.....	329
38.6.1	General.....	329
38.6.2	CIM_ConfigurationReportingService	329
38.6.3	CIM_HostedService	330
39	Software Subprofile	331
39.1	Description	331
39.2	Health and Fault Management Considerations	331
39.3	Cascading Considerations	331
39.4	Supported Subprofiles, and Packages	331
39.5	Methods of the Profile	331
39.6	Client Considerations and Recipes	332
39.7	Registered Name and Version	332
39.8	CIM Elements.....	332
39.8.1	General.....	332
39.8.2	CIM_InstalledSoftwareIdentity.....	332
39.8.3	CIM_SoftwareIdentity	332
40	Software Inventory Profile	334
40.1	Synopsis.....	334
40.2	Description	334
40.2.1	General.....	334
40.2.2	Relationship to the SMI-S Software Profile	334
40.3	Implementation.....	335
40.3.1	General.....	335
40.3.2	Software Installation and Update	335
40.3.3	Health and Fault Management Consideration	335
40.3.4	Cascading Considerations	335
40.4	Methods	335
40.5	Use Cases.....	335
40.6	CIM Elements.....	335
40.6.1	General.....	335
40.6.2	CIM_ElementSoftwareIdentity.....	336
40.6.3	CIM_HostedAccessPoint.....	336
40.6.4	CIM_HostedCollection.....	337
40.6.5	CIM_InstalledSoftwareIdentity.....	337
40.6.6	CIM_MemberOfCollection	337
40.6.7	CIM_OrderedComponent	337
40.6.8	CIM_OrderedDependency	338
40.6.9	CIM_SAPAvailableForElement	338
40.6.10	CIM_SoftwareIdentity.....	338
40.6.11	CIM_SoftwareIdentityResource	339
40.6.12	CIM_SystemSpecificCollection	339
41	Server Profile.....	341
41.1	Description	341
41.1.1	Model Overview.....	341
41.1.2	Use of model fields to Populate the SLP template	341
41.1.3	Support for Indications	343
41.1.4	Security Background	343
41.2	Health and Fault Management.....	343
41.3	Cascading Considerations	343
41.4	Supported Subprofiles and Packages.....	343
41.5	Methods of the Profile	343
41.6	Client Considerations and Recipes – Segregate a SAN Device Type	343
41.7	Registered Name and Version	345
41.8	CIM Elements.....	345

41.8.1	General.....	345
41.8.2	CIM_CIMXMLCommunicationMechanism.....	346
41.8.3	CIM_CommMechanismForManager	346
41.8.4	CIM_HostedAccessPoint.....	347
41.8.5	CIM_HostedService	347
41.8.6	CIM_Namespace.....	347
41.8.7	CIM_NamespaceInManager	348
41.8.8	CIM_ObjectManager	348
41.8.9	CIM_ObjectManagerCommunicationMechanism.....	349
41.8.10	CIM_System.....	350
42	Profile Registration Profile.....	351
42.1	Synopsis.....	351
42.2	Description	351
42.3	Implementation.....	351
42.3.1	General.....	351
42.3.2	ElementConformsToProfile Association.....	352
42.3.3	Associations between Autonomous and Component Profile.....	353
42.3.4	The SMI-S Registered Profile.....	353
42.3.5	Health and Fault Management Consideration.....	355
42.3.6	Cascading Considerations	355
42.4	Methods	355
42.5	Use Cases.....	355
42.5.1	Using the CIM Server Model to Determine SNIA Profiles Supported.....	355
42.5.2	Recipe Assumptions.....	356
42.5.3	Find Servers Supporting a Given Profile	356
42.5.4	Enumerate Profiles Supported by a Given CIM Server	358
42.5.5	Identify the ManagedElement Defined by a Profile	360
42.5.6	Determine the SNIA Version of a Profile	360
42.5.7	Find all Profiles on a Server	361
42.6	CIM Elements.....	362
42.6.1	General.....	362
42.6.2	CIM_ElementConformsToProfile (Associates Domain object (e.g. System) to RegisteredProfile).....	363
42.6.3	CIM_ElementConformsToProfile (Associates RegisteredProfiles for SMI-S and domain profiles).....	363
42.6.4	CIM_ElementSoftwareIdentity (Profile and SW identity)	364
42.6.5	CIM_ElementSoftwareIdentity (Subprofile and SW identity)	364
42.6.6	CIM_Product	364
42.6.7	CIM_ProductSoftwareComponent.....	364
42.6.8	CIM_ReferencedProfile	365
42.6.9	CIM_RegisteredProfile (Domain Registered Profile).....	365
42.6.10	CIM_RegisteredProfile (The SMI-S Registered Profile).....	366
42.6.11	CIM_RegisteredSubProfile.....	367
42.6.12	CIM_SoftwareIdentity.....	367
42.6.13	CIM_SubProfileRequiresProfile	368
43	Indication Profile.....	369
43.1	Description	369
43.1.1	General.....	369
43.1.2	IndicationFilter Names.....	371
43.1.3	Basic Indication Classes and Association	371
43.1.4	Life Cycle Indications	373
43.1.5	AlertIndications.....	374
43.1.6	Indication Delivery	375
43.1.7	Instrumentation Requirements	375

43.2	Health and Fault Management Considerations	379
43.2.1	Elements Reporting Health	379
43.2.2	Health State Transformations and Dependencies	379
43.2.3	Standard Errors Produced	379
43.2.4	Cause and effect associations	379
43.2.5	Indication Correlation	379
43.3	Cascading Considerations	379
43.4	Supported Profiles, Subprofiles and Packages	379
43.5	Methods of the Profile	379
43.5.1	Extrinsic Methods of the Profile	379
43.5.2	Intrinsic Methods of the Profile	379
43.6	Client Considerations and Recipes	383
43.6.1	Use of Profile Specific Recipes	383
43.6.2	General Client Considerations	383
43.6.3	Discovery of Implementation variations	383
43.6.4	Client Defined Filters	384
43.6.5	Creation of IndicationFilter and ListenerDestination Instances	384
43.6.6	Creation of IndicationSubscription Instances	384
43.6.7	Determine if the indication subscription requested already exists	384
43.6.8	Listenable Instance Notification	385
43.6.9	Life Cycle Event Subscription Description	385
43.6.10	Subscription for alert indications	386
43.6.11	Listenable Interface Modification Notification	386
43.6.12	Subscribe for Lifecycle Events where OperationalStatus Changes	386
43.7	Registered Name and Version	387
43.8	CIM Elements	387
43.8.1	General	387
43.8.2	CIM_AlertIndication	387
43.8.3	CIM_IndicationFilter (client defined)	389
43.8.4	CIM_IndicationFilter (pre-defined)	390
43.8.5	CIM_IndicationSubscription	391
43.8.6	CIM_InstCreation	392
43.8.7	CIM_InstDeletion	392
43.8.8	CIM_InstModification	393
43.8.9	CIM_ListenerDestinationCIMXML (Indication Handler)	394
44	Experimental Indication Profile	395
44.1	Description	395
44.1.1	General	395
44.1.2	Basic Indication Classes and Association	396
44.1.3	AlertingManagedElement encoding in AlertIndication Instances	396
44.1.4	Instrumentation Requirements	396
44.1.5	Semi-Fixed Client Specific Indication Filters	399
44.1.6	Filter Collections	400
44.1.7	Indication Configuration Services	403
44.2	Fault Management Considerations	405
44.2.1	Indication Correlation	405
44.3	Cascading Considerations	406
44.4	Supported Profiles, Subprofiles and Packages	406
44.5	Methods of the Profile	407
44.5.1	Extrinsic Methods of the Profile	407
44.5.2	Intrinsic Methods of the Profile	410
44.6	Client Considerations and Recipes	413
44.6.1	General	413
44.6.2	Testing a Listener Destination	413

44.6.3	Discovering predefined IndicationFilters of an implementation	414
44.6.4	Creating a subscription to a predefined IndicationFilter	415
44.6.5	Creating a client defined indication and subscription	415
44.6.6	Creating a semi-fixed indication filter	416
44.6.7	Creating a FilterCollection	416
44.7	Registered Name and Version	417
44.8	CIM Elements.....	417
44.8.1	General.....	417
44.8.2	CIM_AlertIndication	418
44.8.3	CIM_ElementCapabilities (Indication Config Service to Capabilities)	420
44.8.4	CIM_FilterCollection (Client Defined)	420
44.8.5	CIM_FilterCollectionSubscription (Filter Collection Subscription)	420
44.8.6	CIM_HostedCollection (Hosted Filter Collection)	421
44.8.7	CIM_HostedService (Indication Config Service to System)	421
44.8.8	CIM_IndicationFilter (client defined).....	422
44.8.9	CIM_IndicationFilter (pre-defined).....	423
44.8.10	CIM_IndicationSubscription	424
44.8.11	CIM_InstCreation	424
44.8.12	CIM_InstDeletion.....	425
44.8.13	CIM_InstModification.....	426
44.8.14	CIM_ListenerDestinationCIMXML (Indication Handler)	427
44.8.15	CIM_ListenerDestinationWSManagement (WS-Man Indication Handler).....	427
44.8.16	CIM_MemberOfCollection (Filter Collection to Filters).....	428
44.8.17	SNIA_IndicationConfigurationCapabilities	428
44.8.18	SNIA_IndicationConfigurationService.....	429
45	Object Manager Adapter Subprofile	430
45.1	Description	430
45.1.1	General.....	430
45.1.2	Instance Diagram	430
45.2	Health and Fault Management.....	430
45.3	Cascading Considerations	430
45.4	Supported Subprofiles and Packages.....	430
45.5	Methods of the Profile	430
45.6	Client Considerations and Recipes	431
45.7	Registered Name and Version	431
45.8	CIM Elements.....	431
45.8.1	General.....	431
45.8.2	CIM_CommMechanismForObjectManagerAdapter	431
45.8.3	CIM_ObjectManagerAdapter.....	431
46	Proxy Server System Management Subprofile.....	433
46.1	Description	433
46.1.1	General.....	433
46.1.2	Relationship to Server Profile	433
46.1.3	Model.....	433
46.1.4	Creation Considerations.....	434
46.2	Health and Fault Management Consideration.....	435
46.3	Cascading Considerations	435
46.4	Supported Profiles, Subprofiles, and Packages.....	435
46.5	Methods of the Profile	435
46.5.1	General.....	435
46.5.2	AddSystem	435
46.5.3	DiscoverSystems.....	436
46.5.4	RemoveSystem	437
46.6	Client Considerations and Recipes	437

46.6.1	General.....	437
46.6.2	Use Case 1: Add Device	438
46.6.3	Use Case 2: Remove Device	438
46.7	Registered Name and Version	438
46.8	CIM Elements.....	438
46.8.1	General.....	438
46.8.2	CIM_HostedService	439
46.8.3	SNIA_SystemRegistrationCapabilities	439
46.8.4	SNIA_SystemRegistrationService	440
47	Device Credentials Subprofile	441
47.1	Description	441
47.1.1	General.....	441
47.1.2	Instance Diagram	441
47.2	Health and Fault Management Considerations	441
47.3	Cascading Considerations	441
47.4	Supported Subprofiles and Packages.....	441
47.5	Extrinsic Methods of this Profile	441
47.6	Client Considerations and Recipes	441
47.7	Registered Name and Version	441
47.8	CIM Elements.....	442
47.8.1	General.....	442
47.8.2	CIM_HostedService	442
47.8.3	CIM_SharedSecret.....	442
47.8.4	CIM_SharedSecretIsShared	442
47.8.5	CIM_SharedSecretService	443
48	Miscellaneous Security Profiles.....	444
49	Operational Power Profile.....	445
49.1	Synopsis.....	445
49.2	Description	445
49.3	Implementation.....	445
49.3.1	Model Overview.....	445
49.3.2	Element Types	446
49.3.3	Power Metric Attributes	449
49.3.4	Bulk Retrieval	450
49.3.5	Default Manifest Collection.....	450
49.3.6	Client Defined Manifest Collection	450
49.3.7	Capabilities Support for Operational Power Profile	451
49.3.8	Health and Fault Management Consideration	452
49.3.9	Cascading Considerations	452
49.4	Methods of the Profile	452
49.4.1	Extrinsic Methods of the Profile	452
49.4.2	Intrinsic Methods of this Profile	457
49.5	Use Cases – Client Considerations and Recipes	457
49.6	CIM Elements.....	457
49.6.1	General.....	457
49.6.2	CIM_ComputerSystem (Optional Element)	459
49.6.3	CIM_DiskDrive (Optional Element)	459
49.6.4	CIM_ElementCapabilities	459
49.6.5	CIM_ElementStatisticalData (Optional Element Stats)	459
49.6.6	CIM_ElementStatisticalData (Top Level System Stats)	459
49.6.7	CIM_Fan (Optional Element).....	460
49.6.8	CIM_HostedCollection (Client Defined).....	460
49.6.9	CIM_HostedCollection (Default).....	460
49.6.10	CIM_HostedCollection (System to StatisticsCollection).....	461

49.6.11	CIM_HostedService	461
49.6.12	CIM_LogicalModule (Optional Element)	461
49.6.13	CIM_MemberOfCollection (Member of client defined collection)	462
49.6.14	CIM_MemberOfCollection (Member of power statistics collection)	462
49.6.15	CIM_MemberOfCollection (Member of pre-defined collection)	462
49.6.16	CIM_PowerSource (Optional Element)	463
49.6.17	CIM_PowerSupply (Optional Element)	463
49.6.18	CIM_Processor (Optional Element)	463
49.6.19	CIM_StatisticsCollection	463
49.6.20	SNIA_OperationalPowerManifest (Client Defined)	463
49.6.21	SNIA_OperationalPowerManifest (Provider Support)	464
49.6.22	SNIA_OperationalPowerManifestCollection (Client Defined)	465
49.6.23	SNIA_OperationalPowerManifestCollection (Provider Defined)	466
49.6.24	SNIA_OperationalPowerStatisticalData (Optional Element Stats)	466
49.6.25	SNIA_OperationalPowerStatisticalData (Top Level System Stats)	467
49.6.26	SNIA_OperationalPowerStatisticsCapabilities	467
49.6.27	SNIA_OperationalPowerStatisticsService	468
50	Cross Profile Considerations	470
50.1	Overview	470
50.2	HBA model	470
50.3	Switch Model	471
50.3.1	General	471
50.3.2	Recipes	471
50.4	Array Model	475
50.5	Storage Virtualization Model	477
50.6	Fabric Topology (HBA, Switch, Array)	478
50.6.1	General	478
50.6.2	Logical Device Composition	479
Annex A	(informative) SMI-S Information Model	525
Bibliography		527

Figure 1 - Experimental Maturity Level Tag	46
Figure 2 - Implemented Maturity Level Tag	47
Figure 3 - Stable Maturity Level Tag	47
Figure 4 - Deprecated Tag	48
Figure 5 - Generic Target Port Classes	55
Figure 6 - LogicalPort Class Hierarchy	56
Figure 7 - Generic Target with LUN Masking	57
Figure 8 - SPI Target Port Instance Diagram	61
Figure 9 - FC Target Port Instance Diagram	67
Figure 10 - iSCSI Target Ports Subprofile Instance Diagram	74
Figure 11 - Serial Attached SCSI (SAS) Target Port Instance Diagram	110
Figure 12 - SATA Target Port Instance Diagram	117
Figure 13 - SB Target Port Instance Diagram	122
Figure 14 - DA Port Instance Diagram	128
Figure 15 - Generic Initiator Port Model	132
Figure 16 - Optional Connectivity Collection Model	133
Figure 17 - Optional Full-Path Model	133
Figure 18 - HBA and Disk Model	134
Figure 19 - HBA and Tape or Optical Devices	135
Figure 20 - Port Statistics	135
Figure 21 - Port Statistics Hierarchy	136
Figure 22 - SPI Initiator Port Instance Diagram	143
Figure 23 - iSCSI Initiator Port Instance Diagram	153
Figure 24 - Fibre Channel Initiator Instance Diagram	162
Figure 25 - FC Node Model	162
Figure 26 - SAS Initiator Port Model	174
Figure 27 - ATA Initiator Port Class Diagram	185
Figure 28 - Fibre Channel Initiator Instance Diagram	194
Figure 29 - FCoE Initiator Instance Diagram	204
Figure 30 - Optional Target Element Model	205
Figure 31 - Logical Port Group Model	206
Figure 32 - System-wide Remote Access Point	218
Figure 33 - Access Point Instance Diagram	219
Figure 34 - Instance Diagram for Logical Topology	223
Figure 35 - Resource Allocation/Deallocation Instance Diagram	225
Figure 36 - Cascading Server Topology	226
Figure 37 - Instance Diagram for Cascading with Resource Ownership	227
Figure 38 - Instance Diagram for Cascading with Credential Management Subprofile	228
Figure 39 - Modeling of Cascading Capabilities	229
Figure 40 - Job Control Subprofile Model	257
Figure 41 - Storage Configuration	262
Figure 42 - Location Instance	268
Figure 43 - Two Redundant Systems Instance Diagram	272
Figure 44 - Multiple Redundancy Tier Instance Diagram	273
Figure 45 - System Level Numbers	274
Figure 46 - Physical Package Package Mandatory Classes	282
Figure 47 - Modeling for well defined subcomponents	283
Figure 48 - Physical Package Package with Optional Classes	284
Figure 49 - Media Access Device Class Diagram	317

Figure 50 - Enclosure with Two Arrays	325
Figure 51 - Model for Disk in Enclosure	328
Figure 52 - Software Instance Diagram	331
Figure 53 - Server Model	341
Figure 54 - Profile Registration Model	352
Figure 55 - Associations between RegisteredProfile instances	353
Figure 56 - Model for SMI-S Registered Profile	354
Figure 57 - Model for Provider Versions	354
Figure 58 - Indication Profile and Namespaces	369
Figure 59 - Indication Profile Instance Diagram	372
Figure 60 - Indication Profile Instance Diagram	396
Figure 61 - Anatomy of IndicationIdentifier	397
Figure 62 - Predefined Filter Collections	401
Figure 63 - Client Defined Filter Collections	402
Figure 64 - Indication Configuration Service Classes	404
Figure 65 - ObjectManagerAdapter Subprofile Model	430
Figure 66 - Proxy Server System Management Model	434
Figure 67 - DeviceCredentials Subprofile Model	441
Figure 68 - Operational Power Profile Summary	446
Figure 69 - Model for Element Types	447
Figure 70 - Classes related to Top-level System Power Statistics	448
Figure 71 - System Diagram	470
Figure 72 - Host Bus Adapter Model	470
Figure 73 - Switch Model	471
Figure 74 - Array Instance	476
Figure 75 - Virtualization Instance	477
Figure 76 - Fabric Topology	478

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

Table 1 - Profile Components	44
Table 2 - Modeling of Common Storage Devices in CIM	57
Table 3 - CIM Elements for Generic Target Ports	58
Table 4 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	58
Table 5 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint	59
Table 6 - SMI Referenced Properties/Methods for CIM_LogicalPort	59
Table 7 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint	60
Table 8 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port)	60
Table 9 - Related Profiles for SPI Target Ports	61
Table 10 - SPIPort OperationalStatus	62
Table 11 - CIM Elements for SPI Target Ports	62
Table 12 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	63
Table 13 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint	63
Table 14 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint	64
Table 15 - SMI Referenced Properties/Methods for CIM_SPIPort	64
Table 16 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port)	65
Table 17 - Related Profiles for FC Target Ports	66
Table 18 - FCPort OperationalStatus	67
Table 19 - CIM Elements for FC Target Ports	68
Table 20 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	69
Table 21 - SMI Referenced Properties/Methods for CIM_FCPort	69
Table 22 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint	70
Table 23 - SMI Referenced Properties/Methods for CIM_ProtocolControllerForPort	70
Table 24 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint	71
Table 25 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port)	71
Table 26 - iSCSI Terminology and SMI-S Class Names	72
Table 27 - EthernetPort OperationalStatus	75
Table 28 - CIM Elements for iSCSI Target Ports	90
Table 29 - SMI Referenced Properties/Methods for CIM_BindsTo (TCPProtocolEndpoint to IPProtocolEndpoint)	92
Table 30 - SMI Referenced Properties/Methods for CIM_BindsTo (iSCSIProtocolEndpoint to TCPProtocolEndpoint)	93
Table 31 - SMI Referenced Properties/Methods for CIM_ConcreteDependency	93
Table 32 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation (EthernetPort to IPProtocolEndpoint)	93
Table 33 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation (EthernetPort to iSCSIProtocolEndpoint)	94
Table 34 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (iSCSIConfigurationCapabilities to System)	94
Table 35 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (iSCSIConfigurationCapabilities to iSCSIConfigurationService)	94
Table 36 - SMI Referenced Properties/Methods for CIM_ElementSettingData (iSCSIConnectionSettings to TCPProtocolEndpoint)	95
Table 37 - SMI Referenced Properties/Methods for CIM_ElementSettingData (iSCSIConnectionSettings to iSCSIProtocolEndpoint)	95
Table 38 - SMI Referenced Properties/Methods for CIM_ElementSettingData (iSCSIConnectionSettings to iSCSIProtocolController)	95
Table 39 - SMI Referenced Properties/Methods for CIM_ElementSettingData (iSCSIConnectionSettings to System)	96
Table 40 - SMI Referenced Properties/Methods for CIM_ElementSettingData (iSCSIConnectionSettings to iSCSIProtocolEndpoint)	96
Table 41 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (iSCSIConnectionStatistics to SCSIProtocolController)	96
Table 42 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (iSCSIConnectionFailures to SCSIProtocolController)	97

Table 43 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (iSCSIStatistics to iSCSIStatistics)	97
Table 44 - SMI Referenced Properties/Methods for CIM_EndpointOfNetworkPipe (iSCSIConnection to TCPProtocolEndpoint)	97
Table 45 - SMI Referenced Properties/Methods for CIM_EndpointOfNetworkPipe (iSCSIStatistics to iSCSIProtocolEndpoint)	98
Table 46 - SMI Referenced Properties/Methods for CIM_EthernetPort	98
Table 47 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (System to IPProtocolEndpoint)	99
Table 48 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (System to TCPProtocolEndpoint)	99
Table 49 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (System to iSCSIProtocolEndpoint)	99
Table 50 - SMI Referenced Properties/Methods for CIM_HostedCollection	100
Table 51 - SMI Referenced Properties/Methods for CIM_HostedService	100
Table 52 - SMI Referenced Properties/Methods for CIM_IPProtocolEndpoint	100
Table 53 - SMI Referenced Properties/Methods for CIM_MemberOfCollection	101
Table 54 - SMI Referenced Properties/Methods for CIM_NetworkPipeComposition	101
Table 55 - SMI Referenced Properties/Methods for CIM_SAPAvailableForElement	101
Table 56 - SMI Referenced Properties/Methods for CIM_SCSIProtocolController	102
Table 57 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to EthernetPort)	102
Table 58 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to SCSIProtocolController)	102
Table 59 - SMI Referenced Properties/Methods for CIM_SystemSpecificCollection	103
Table 60 - SMI Referenced Properties/Methods for CIM_TCPProtocolEndpoint	103
Table 61 - SMI Referenced Properties/Methods for CIM_iSCSICapabilities	103
Table 62 - SMI Referenced Properties/Methods for CIM_iSCSIConfigurationCapabilities	104
Table 63 - SMI Referenced Properties/Methods for CIM_iSCSIConfigurationService	104
Table 64 - SMI Referenced Properties/Methods for CIM_iSCSIConnection	105
Table 65 - SMI Referenced Properties/Methods for CIM_iSCSIConnectionSettings	105
Table 66 - SMI Referenced Properties/Methods for CIM_iSCSILoginStatistics	106
Table 67 - SMI Referenced Properties/Methods for CIM_iSCSIProtocolEndpoint	107
Table 68 - SMI Referenced Properties/Methods for CIM_iSCSIStatistics	107
Table 69 - SMI Referenced Properties/Methods for CIM_iSCSIStatisticsFailures	108
Table 70 - SMI Referenced Properties/Methods for CIM_iSCSIStatisticsSettings	108
Table 71 - SMI Referenced Properties/Methods for CIM_iSCSIStatisticsStatistics	109
Table 72 - Related Profiles for SAS Target Ports	110
Table 73 - SASPort OperationalStatus	111
Table 74 - CIM Elements for SAS Target Ports	112
Table 75 - SMI Referenced Properties/Methods for CIM_ConcreteComponent	112
Table 76 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	113
Table 77 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint	113
Table 78 - SMI Referenced Properties/Methods for CIM_SASPort	114
Table 79 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint	114
Table 80 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port)	115
Table 81 - SMI Referenced Properties/Methods for CIM_SystemDevice (SAS PHY)	115
Table 82 - SMI Referenced Properties/Methods for SNIA_SASPHY	115
Table 83 - Related Profiles for SATA Target Ports	116
Table 84 - ATAPort OperationalStatus	117
Table 85 - CIM Elements for SATA Target Ports	118
Table 86 - SMI Referenced Properties/Methods for CIM_ATAPort	118
Table 87 - SMI Referenced Properties/Methods for CIM_ATAPortProtocolEndpoint	119
Table 88 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	119

Table 89 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint.....	120
Table 90 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port).....	120
Table 91 - Related Profiles for SB Target Ports.....	121
Table 92 - FCPort OperationalStatus.....	122
Table 93 - CIM Elements for SB Target Ports.....	123
Table 94 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	124
Table 95 - SMI Referenced Properties/Methods for CIM_FCPort.....	124
Table 96 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint.....	125
Table 97 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port).....	125
Table 98 - SMI Referenced Properties/Methods for SNIA_SBProtocolEndpoint.....	126
Table 99 - CIM Elements for DA Target Ports	129
Table 100 - SMI Referenced Properties/Methods for CIM_DAPort	129
Table 101 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	130
Table 102 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint.....	130
Table 103 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint	131
Table 104 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port)	131
Table 105 - CIM Elements for Generic Initiator Ports	137
Table 106 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection.....	137
Table 107 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	138
Table 108 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)	138
Table 109 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator).....	138
Table 110 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target).....	139
Table 111 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)	139
Table 112 - SMI Referenced Properties/Methods for CIM_LogicalPort.....	140
Table 113 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Connectivity Collection)	140
Table 114 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint (Initiator)	140
Table 115 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint (Target).....	141
Table 116 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)	142
Table 117 - SMI Referenced Properties/Methods for SNIA_LogicalPortStatistics.....	142
Table 118 - SPIPort OperationalStatus.....	143
Table 119 - CIM Elements for SPI Initiator Ports	144
Table 120 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection.....	145
Table 121 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	146
Table 122 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)	146
Table 123 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator).....	146
Table 124 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target).....	147
Table 125 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)	147
Table 126 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Connectivity Collection)	148
Table 127 - SMI Referenced Properties/Methods for CIM_SCSIInitiatorTargetLogicalUnitPath	148
Table 128 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Initiator)	148
Table 129 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Target)	149
Table 130 - SMI Referenced Properties/Methods for CIM_SPIPort.....	150
Table 131 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)	150
Table 132 - SMI Referenced Properties/Methods for SNIA_LogicalPortStatistics.....	151
Table 133 - Related Profiles for iSCSI Initiator Ports	152
Table 134 - EthernetPort OperationalStatus.....	153
Table 135 - CIM Elements for iSCSI Initiator Ports.....	154
Table 136 - SMI Referenced Properties/Methods for CIM_BindsTo (Host Hardware RAID Controller)	155
Table 137 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation (IPProtocolEndpoint to EthernetPort).....	155

Table 138 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation (iSSIPProtocolEndpoint to EthernetPort)	156
Table 139 - SMI Referenced Properties/Methods for CIM_EthernetPort (Host Hardware RAID Controller)	156
Table 140 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (System to IPProtocolEndpoint)	156
Table 141 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (System to TCPProtocolEndpoint)	157
Table 142 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (System to iSCSIProtocolEndpoint)	157
Table 143 - SMI Referenced Properties/Methods for CIM_IPProtocolEndpoint (Host Hardware RAID Controller)	157
Table 144 - SMI Referenced Properties/Methods for CIM_LogicalDevice (Host Hardware RAID Controller)	158
Table 145 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to EthernetPort)	158
Table 146 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to LogicalDevice)	159
Table 147 - SMI Referenced Properties/Methods for CIM_TCPProtocolEndpoint (Host Hardware RAID Controller)	159
Table 148 - SMI Referenced Properties/Methods for CIM_iSCSIProtocolEndpoint (Host Hardware RAID Controller)	160
Table 149 - Related Profiles for FC Initiator Ports	161
Table 150 - FCPort OperationalStatus	163
Table 151 - CIM Elements for FC Initiator Ports	164
Table 152 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection	165
Table 153 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	166
Table 154 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)	166
Table 155 - SMI Referenced Properties/Methods for CIM_FCPort	167
Table 156 - SMI Referenced Properties/Methods for CIM_FCPortStatistics	168
Table 157 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator)	169
Table 158 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target)	169
Table 159 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)	169
Table 160 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Connectivity Collection)	170
Table 161 - SMI Referenced Properties/Methods for CIM_ProtocolControllerForPort	170
Table 162 - SMI Referenced Properties/Methods for CIM_SCSIInitiatorTargetLogicalUnitPath	171
Table 163 - SMI Referenced Properties/Methods for CIM_SCSIProtocolController	171
Table 164 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Initiator)	172
Table 165 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Target)	172
Table 166 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)	173
Table 167 - SASPort OperationalStatus	175
Table 168 - CIM Elements for SAS Initiator Ports	175
Table 169 - SMI Referenced Properties/Methods for CIM_ATAProtocolEndpoint (Initiator)	176
Table 170 - SMI Referenced Properties/Methods for CIM_BindsTo	177
Table 171 - SMI Referenced Properties/Methods for CIM_ConcreteComponent	177
Table 172 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection	177
Table 173 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	178
Table 174 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (PHY Statistics)	178
Table 175 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)	178
Table 176 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator)	179
Table 177 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target)	179
Table 178 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)	179
Table 179 - SMI Referenced Properties/Methods for CIM_MemberOfCollection	

(Connectivity Collection)	180
Table 180 - SMI Referenced Properties/Methods for CIM_SASPort	180
Table 181 - SMI Referenced Properties/Methods for CIM_SCSIInitiatorTargetLogicalUnitPath	181
Table 182 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Initiator)	181
Table 183 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Target)	182
Table 184 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator PHY)	182
Table 185 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)	183
Table 186 - SMI Referenced Properties/Methods for SNIA_LogicalPortStatistics	183
Table 187 - SMI Referenced Properties/Methods for SNIA_SASPHY	183
Table 188 - SMI Referenced Properties/Methods for SNIA_SASPhyStatistics	184
Table 189 - ATAPort OperationalStatus	186
Table 190 - CIM Elements for ATA Initiator Ports	187
Table 191 - SMI Referenced Properties/Methods for CIM_ATAInitiatorTargetLogicalUnitPath	188
Table 192 - SMI Referenced Properties/Methods for CIM_ATAPort	188
Table 193 - SMI Referenced Properties/Methods for CIM_ATAProtocolEndpoint (Initiator)	189
Table 194 - SMI Referenced Properties/Methods for CIM_ATAProtocolEndpoint (Target)	189
Table 195 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection	190
Table 196 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	190
Table 197 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)	191
Table 198 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator)	191
Table 199 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target)	191
Table 200 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)	192
Table 201 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Connectivity Collection)	192
Table 202 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)	192
Table 203 - SMI Referenced Properties/Methods for SNIA_LogicalPortStatistics	193
Table 204 - Related Profiles for SB Initiator Ports	194
Table 205 - FCPort OperationalStatus	195
Table 206 - CIM Elements for SB Initiator Ports	196
Table 207 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection	197
Table 208 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	197
Table 209 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)	197
Table 210 - SMI Referenced Properties/Methods for CIM_FCPort	198
Table 211 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator)	199
Table 212 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target)	199
Table 213 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)	199
Table 214 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Connectivity Collection)	200
Table 215 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)	200
Table 216 - SMI Referenced Properties/Methods for SNIA_LogicalPortStatistics	201
Table 217 - SMI Referenced Properties/Methods for SNIA_SBInitiatorTargetLogicalUnitPath	201
Table 218 - SMI Referenced Properties/Methods for SNIA_SBProtocolEndpoint (Initiator)	202
Table 219 - SMI Referenced Properties/Methods for SNIA_SBProtocolEndpoint (Target)	202
Table 220 - FCPort OperationalStatus	206
Table 221 - CIM Elements for FCoE Initiator Ports	207
Table 222 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection	208
Table 223 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation	208
Table 224 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)	209
Table 225 - SMI Referenced Properties/Methods for CIM_EthernetPort	209
Table 226 - SMI Referenced Properties/Methods for CIM_FCPort	210
Table 227 - SMI Referenced Properties/Methods for CIM_FCPortStatistics	211
Table 228 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator)	212

Table 229 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target)	212
Table 230 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)	212
Table 231 - SMI Referenced Properties/Methods for CIM_HostedCollection (FC Node)	213
Table 232 - SMI Referenced Properties/Methods for CIM_HostedDependency (NetworkPort to FCPort)	213
Table 233 - SMI Referenced Properties/Methods for CIM_LogicalPortGroup	213
Table 234 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Connectivity Collection)	214
Table 235 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (FC Node)	214
Table 236 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint (Initiator)	214
Table 237 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint (Target)	215
Table 238 - SMI Referenced Properties/Methods for CIM_SCSIInitiatorTargetLogicalUnitPath	215
Table 239 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Initiator)	216
Table 240 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Target)	216
Table 241 - SMI Referenced Properties/Methods for CIM_SystemDevice (Ethernet Port)	217
Table 242 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)	217
Table 243 - RemoteAccessPoint InfoFormat and AccessInfo Properties	219
Table 244 - CIM Elements for Access Points	220
Table 245 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint	220
Table 246 - SMI Referenced Properties/Methods for CIM_RemoteServiceAccessPoint	221
Table 247 - SMI Referenced Properties/Methods for CIM_SAPAvailableForElement	221
Table 248 - Supported Profiles for Cascading	230
Table 249 - Extrinsic Methods Supported by Cascading Subprofile	230
Table 250 - Cascading Capabilities Patterns	232
Table 251 - CIM Elements for Cascading	233
Table 252 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Leaf System)	235
Table 253 - SMI Referenced Properties/Methods for CIM_Dependency (Object Managers)	236
Table 254 - SMI Referenced Properties/Methods for CIM_Dependency (Profile to Object Manager)	236
Table 255 - SMI Referenced Properties/Methods for CIM_Dependency (Systems)	236
Table 256 - SMI Referenced Properties/Methods for CIM_ElementCapabilities	237
Table 257 - SMI Referenced Properties/Methods for CIM_ElementConformsToProfile (Leaf)	237
Table 258 - SMI Referenced Properties/Methods for CIM_HostedCollection (Allocated Resources)	238
Table 259 - SMI Referenced Properties/Methods for CIM_HostedCollection (Remote Resources)	238
Table 260 - SMI Referenced Properties/Methods for CIM_HostedService (Allocation Service)	238
Table 261 - SMI Referenced Properties/Methods for CIM_HostedService (Object Manager)	239
Table 262 - SMI Referenced Properties/Methods for CIM_LogicalDisk	239
Table 263 - SMI Referenced Properties/Methods for CIM_LogicalIdentity (General)	240
Table 264 - SMI Referenced Properties/Methods for CIM_LogicalIdentity (LogicalDisk)	241
Table 265 - SMI Referenced Properties/Methods for CIM_LogicalIdentity (StorageVolume)	241
Table 266 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Allocated Resources)	242
Table 267 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Remote Resources)	242
Table 268 - SMI Referenced Properties/Methods for CIM_Namespace (Leaf)	242
Table 269 - SMI Referenced Properties/Methods for CIM_NamespaceInManager (Leaf)	243
Table 270 - SMI Referenced Properties/Methods for CIM_ObjectManager (Leaf)	243
Table 271 - SMI Referenced Properties/Methods for CIM_RegisteredProfile (Leaf)	244
Table 272 - SMI Referenced Properties/Methods for CIM_RemoteServiceAccessPoint (Leaf)	244
Table 273 - SMI Referenced Properties/Methods for CIM_SAPAvailableForElement	245
Table 274 - SMI Referenced Properties/Methods for CIM_StorageVolume	245
Table 275 - SMI Referenced Properties/Methods for CIM_SystemDevice (Leaf Devices)	246

Table 276 - SMI Referenced Properties/Methods for SNIA_AllocatedResources	247
Table 277 - SMI Referenced Properties/Methods for SNIA_AllocationService.....	247
Table 278 - SMI Referenced Properties/Methods for SNIA_CascadingCapabilities.....	248
Table 279 - SMI Referenced Properties/Methods for SNIA_RemoteResources	249
Table 280 - OperationalStatus Details	252
Table 281 - CIM Elements for Health	254
Table 282 - SMI Referenced Properties/Methods for CIM_ComputerSystem	255
Table 283 - SMI Referenced Properties/Methods for CIM_LogicalDevice.....	255
Table 284 - SMI Referenced Properties/Methods for CIM_RelatedElementCausingError	256
Table 285 - OperationalStatus to Job State Mapping	259
Table 286 - Standard Message for Job Control Subprofile	260
Table 287 - CIM Elements for Job Control	263
Table 288 - SMI Referenced Properties/Methods for CIM_AffectedJobElement.....	264
Table 289 - SMI Referenced Properties/Methods for CIM_AssociatedJobMethodResult	264
Table 290 - SMI Referenced Properties/Methods for CIM_ConcreteJob.....	265
Table 291 - SMI Referenced Properties/Methods for CIM_MethodResult.....	267
Table 292 - SMI Referenced Properties/Methods for CIM_OwningJobElement.....	267
Table 293 - CIM Elements for Location.....	269
Table 294 - SMI Referenced Properties/Methods for CIM_Location	269
Table 295 - SMI Referenced Properties/Methods for CIM_PhysicalElementLocation	269
Table 296 - Redundancy Type.....	273
Table 297 - Supported Profiles for Multiple Computer System	276
Table 298 - CIM Elements for Multiple Computer System	279
Table 299 - SMI Referenced Properties/Methods for CIM_ComponentCS	280
Table 300 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Non-Top-Level System).....	280
Table 301 - SMI Referenced Properties/Methods for CIM_ConcretelIdentity	280
Table 302 - SMI Referenced Properties/Methods for CIM_IsSpare.....	281
Table 303 - SMI Referenced Properties/Methods for CIM_MemberOfCollection	281
Table 304 - SMI Referenced Properties/Methods for CIM_RedundancySet	281
Table 305 - CIM Elements for Physical Package.....	285
Table 306 - SMI Referenced Properties/Methods for CIM_Container	286
Table 307 - SMI Referenced Properties/Methods for CIM_LogicalIdentity	286
Table 308 - SMI Referenced Properties/Methods for CIM_PhysicalElementLocation	286
Table 309 - SMI Referenced Properties/Methods for CIM_PhysicalPackage (Component).....	287
Table 310 - SMI Referenced Properties/Methods for CIM_PhysicalPackage (System)	287
Table 311 - SMI Referenced Properties/Methods for CIM_Product (Component).....	288
Table 312 - SMI Referenced Properties/Methods for CIM_Product (System)	288
Table 313 - SMI Referenced Properties/Methods for CIM_ProductParentChild	289
Table 314 - SMI Referenced Properties/Methods for CIM_ProductPhysicalComponent (Component).....	289
Table 315 - SMI Referenced Properties/Methods for CIM_ProductPhysicalComponent (System)	290
Table 316 - SMI Referenced Properties/Methods for CIM_SystemPackaging (Component)	290
Table 317 - SMI Referenced Properties/Methods for CIM_SystemPackaging (System).....	290
Table 318 - Related Profiles for Power Supply	291
Table 319 - CIM Elements for Power Supply	292
Table 320 - SMI Referenced Properties/Methods for CIM_ElementCapabilities	293
Table 321 - SMI Referenced Properties/Methods for CIM_EnabledLogicalElementCapabilities.....	293
Table 322 - SMI Referenced Properties/Methods for CIM_IsSpare.....	293
Table 323 - SMI Referenced Properties/Methods for CIM_MemberOfCollection	294
Table 324 - SMI Referenced Properties/Methods for CIM_OwningCollectionElement.....	294
Table 325 - SMI Referenced Properties/Methods for CIM_PowerSupply	295
Table 326 - SMI Referenced Properties/Methods for CIM_RedundancySet	295

Table 327 - SMI Referenced Properties/Methods for CIM_SuppliesPower	296
Table 328 - SMI Referenced Properties/Methods for CIM_SystemDevice	296
Table 329 - Related Profiles for Fan	297
Table 330 - CIM Elements for Fan	298
Table 331 - SMI Referenced Properties/Methods for CIM_AssociatedCooling	299
Table 332 - SMI Referenced Properties/Methods for CIM_AssociatedSensor	299
Table 333 - SMI Referenced Properties/Methods for CIM_ElementCapabilities	299
Table 334 - SMI Referenced Properties/Methods for CIM_EnabledLogicalElementCapabilities	300
Table 335 - SMI Referenced Properties/Methods for CIM_Fan	300
Table 336 - SMI Referenced Properties/Methods for CIM_IsSpare	301
Table 337 - SMI Referenced Properties/Methods for CIM_MemberOfCollection	301
Table 338 - SMI Referenced Properties/Methods for CIM_NumericSensor	301
Table 339 - SMI Referenced Properties/Methods for CIM_OwningCollectionElement	302
Table 340 - SMI Referenced Properties/Methods for CIM_RedundancySet	302
Table 341 - SMI Referenced Properties/Methods for CIM_Sensor	303
Table 342 - SMI Referenced Properties/Methods for CIM_SystemDevice	303
Table 343 - CIM Elements for Sensors	304
Table 344 - SMI Referenced Properties/Methods for CIM_AssociatedSensor	305
Table 345 - SMI Referenced Properties/Methods for CIM_ElementCapabilities	306
Table 346 - SMI Referenced Properties/Methods for CIM_EnabledLogicalElementCapabilities	306
Table 347 - SMI Referenced Properties/Methods for CIM_NumericSensor	306
Table 348 - SMI Referenced Properties/Methods for CIM_Sensor	308
Table 349 - SMI Referenced Properties/Methods for CIM_SystemDevice	309
Table 350 - Related Profiles for Base Server	310
Table 351 - CIM Elements for Base Server	311
Table 352 - SMI Referenced Properties/Methods for CIM_ComputerSystem	312
Table 353 - SMI Referenced Properties/Methods for CIM_ComputerSystemPackage	313
Table 354 - SMI Referenced Properties/Methods for CIM_ElementCapabilities	313
Table 355 - SMI Referenced Properties/Methods for CIM_EnabledLogicalElementCapabilities	313
Table 356 - SMI Referenced Properties/Methods for CIM_HostedService	314
Table 357 - SMI Referenced Properties/Methods for CIM_PhysicalPackage	314
Table 358 - SMI Referenced Properties/Methods for CIM_ServiceAffectsElement	315
Table 359 - SMI Referenced Properties/Methods for CIM_TimeService	315
Table 360 - Related Profiles for Media Access Device	316
Table 361 - OperationalStatus For MediaAccessDevice	317
Table 362 - CIM Elements for Media Access Device	318
Table 363 - SMI Referenced Properties/Methods for CIM_EnabledLogicalElementCapabilities	319
Table 364 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint	319
Table 365 - SMI Referenced Properties/Methods for CIM_MediaAccessDevice	319
Table 366 - SMI Referenced Properties/Methods for CIM_PhysicalPackage	320
Table 367 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint	321
Table 368 - SMI Referenced Properties/Methods for CIM_Realizes	321
Table 369 - SMI Referenced Properties/Methods for CIM_SAPAvailableForElement	321
Table 370 - SMI Referenced Properties/Methods for CIM_SystemDevice	322
Table 371 - Related Profiles for Storage Enclosure	323
Table 372 - CIM Elements for Storage Enclosure	329
Table 373 - SMI Referenced Properties/Methods for CIM_ConfigurationReportingService	329
Table 374 - SMI Referenced Properties/Methods for CIM_HostedService	330
Table 375 - CIM Elements for Software	332
Table 376 - SMI Referenced Properties/Methods for CIM_InstalledSoftwareIdentity	332
Table 377 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity	333
Table 378 - Related Profiles for Software Inventory	334
Table 379 - CIM Elements for Software Inventory	335

Table 380 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity	336
Table 381 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint	336
Table 382 - SMI Referenced Properties/Methods for CIM_HostedCollection	337
Table 383 - SMI Referenced Properties/Methods for CIM_InstalledSoftwareIdentity	337
Table 384 - SMI Referenced Properties/Methods for CIM_MemberOfCollection	337
Table 385 - SMI Referenced Properties/Methods for CIM_OrderedComponent	338
Table 386 - SMI Referenced Properties/Methods for CIM_OrderedDependency	338
Table 387 - SMI Referenced Properties/Methods for CIM_SAPAvailableForElement	338
Table 388 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity	339
Table 389 - SMI Referenced Properties/Methods for CIM_SoftwareIdentityResource	339
Table 390 - SMI Referenced Properties/Methods for CIM_SystemSpecificCollection	340
Table 391 - Supported Profiles for Server	343
Table 392 - CIM Elements for Server	345
Table 393 - SMI Referenced Properties/Methods for CIM_CIMXMLCommunicationMechanism	346
Table 394 - SMI Referenced Properties/Methods for CIM_CommMechanismForManager	347
Table 395 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint	347
Table 396 - SMI Referenced Properties/Methods for CIM_HostedService	347
Table 397 - SMI Referenced Properties/Methods for CIM_Namespace	348
Table 398 - SMI Referenced Properties/Methods for CIM_NamespaceInManager	348
Table 399 - SMI Referenced Properties/Methods for CIM_ObjectManager	349
Table 400 - SMI Referenced Properties/Methods for CIM_ObjectManagerCommunicationMechanism	349
Table 401 - SMI Referenced Properties/Methods for CIM_System	350
Table 402 - CIM Elements for Profile Registration	362
Table 403 - SMI Referenced Properties/Methods for CIM_ElementConformsToProfile (Associates Domain object (e.g. System) to RegisteredProfile)	363
Table 404 - SMI Referenced Properties/Methods for CIM_ElementConformsToProfile (Associates RegisteredProfiles for SMI-S and domain profiles)	363
Table 405 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity (Profile and SW identity)	364
Table 406 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity (Subprofile and SW identity)	364
Table 407 - SMI Referenced Properties/Methods for CIM_Product	364
Table 408 - SMI Referenced Properties/Methods for CIM_ProductSoftwareComponent	365
Table 409 - SMI Referenced Properties/Methods for CIM_ReferencedProfile	365
Table 410 - SMI Referenced Properties/Methods for CIM_RegisteredProfile (Domain Registered Profile)	366
Table 411 - SMI Referenced Properties/Methods for CIM_RegisteredProfile (The SMI-S Registered Profile)	366
Table 412 - SMI Referenced Properties/Methods for CIM_RegisteredSubProfile	367
Table 413 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity	368
Table 414 - SMI Referenced Properties/Methods for CIM_SubProfileRequiresProfile	368
Table 415 - Indication Profile Methods that cause Instance Creation, Deletion or Modification	380
Table 416 - CIM Elements for Indication	387
Table 417 - SMI Referenced Properties/Methods for CIM_AlertIndication	388
Table 418 - SMI Referenced Properties/Methods for CIM_IndicationFilter (client defined)	389
Table 419 - SMI Referenced Properties/Methods for CIM_IndicationFilter (pre-defined)	390
Table 420 - SMI Referenced Properties/Methods for CIM_IndicationSubscription	391
Table 421 - SMI Referenced Properties/Methods for CIM_InstCreation	392
Table 422 - SMI Referenced Properties/Methods for CIM_InstDeletion	393
Table 423 - SMI Referenced Properties/Methods for CIM_InstModification	393
Table 424 - SMI Referenced Properties/Methods for CIM_ListenerDestinationCIMXML (Indication Handler)	394
Table 425 - Test that a Listener Destination is Functioning Properly	414
Table 426 - Discovery of Predefined IndicationFilters	414

Table 427 - Create a subscription to a predefined indication filter	415
Table 428 - Create an IndicationFilter and subscribe to it.....	415
Table 429 - Creation of a semi-fixed Indication filters	416
Table 430 - Creation of a client defined FilterCollection	416
Table 431 - CIM Elements for Experimental Indication.....	417
Table 432 - SMI Referenced Properties/Methods for CIM_AlertIndication	419
Table 433 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (Indication Config Service to Capabilities)	420
Table 434 - SMI Referenced Properties/Methods for CIM_FilterCollection (Client Defined)	420
Table 435 - SMI Referenced Properties/Methods for CIM_FilterCollectionSubscription (Filter Collection Subscription)	421
Table 436 - SMI Referenced Properties/Methods for CIM_HostedCollection (Hosted Filter Collection)	421
Table 437 - SMI Referenced Properties/Methods for CIM_HostedService (Indication Config Service to System).....	422
Table 438 - SMI Referenced Properties/Methods for CIM_IndicationFilter (client defined)	422
Table 439 - SMI Referenced Properties/Methods for CIM_IndicationFilter (pre-defined)	423
Table 440 - SMI Referenced Properties/Methods for CIM_IndicationSubscription	424
Table 441 - SMI Referenced Properties/Methods for CIM_InstCreation.....	425
Table 442 - SMI Referenced Properties/Methods for CIM_InstDeletion	425
Table 443 - SMI Referenced Properties/Methods for CIM_InstModification	426
Table 444 - SMI Referenced Properties/Methods for CIM_ListenerDestinationCIMXML (Indication Handler).....	427
Table 445 - SMI Referenced Properties/Methods for CIM_ListenerDestinationWSManagement (WS-Man Indi- cation Handler).....	428
Table 446 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Filter Collection to Filters)	428
Table 447 - SMI Referenced Properties/Methods for SNIA_IndicationConfigurationCapabilities	429
Table 448 - SMI Referenced Properties/Methods for SNIA_IndicationConfigurationService	429
Table 449 - CIM Elements for Object Manager Adapter	431
Table 450 - SMI Referenced Properties/Methods for CIM_CommMechanismForObjectManagerAdapter	431
Table 451 - SMI Referenced Properties/Methods for CIM_ObjectManagerAdapter	432
Table 452 - Capabilities	434
Table 453 - AddSystem Method Parameters	435
Table 454 - AddSystem Return Codes	436
Table 455 - DiscoverSystem Parameters	436
Table 456 - DiscoverSystem Return Codes	437
Table 457 - RemoveSystem Parameters	437
Table 458 - CIM Elements for Proxy Server System Management	438
Table 459 - SMI Referenced Properties/Methods for CIM_HostedService.....	439
Table 460 - SMI Referenced Properties/Methods for SNIA_SystemRegistrationCapabilities	439
Table 461 - SMI Referenced Properties/Methods for SNIA_SystemRegistrationService	440
Table 462 - CIM Elements for Device Credentials	442
Table 463 - SMI Referenced Properties/Methods for CIM_HostedService.....	442
Table 464 - SMI Referenced Properties/Methods for CIM_SharedSecret.....	442
Table 465 - SMI Referenced Properties/Methods for CIM_SharedSecretIsShared.....	443
Table 466 - SMI Referenced Properties/Methods for CIM_SharedSecretService	443
Table 467 - Related Profiles for Operational Power.....	445
Table 468 - Creation, Deletion and Modification Methods	452
Table 469 - CIM Elements for Operational Power	457
Table 470 - SMI Referenced Properties/Methods for CIM_ElementCapabilities	459
Table 471 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Optional Element Stats).....	459
Table 472 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData	

(Top Level System Stats).....	460
Table 473 - SMI Referenced Properties/Methods for CIM_HostedCollection (Client Defined).....	460
Table 474 - SMI Referenced Properties/Methods for CIM_HostedCollection (Default)	461
Table 475 - SMI Referenced Properties/Methods for CIM_HostedCollection (System to StatisticsCollection)	461
Table 476 - SMI Referenced Properties/Methods for CIM_HostedService.....	461
Table 477 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Member of client defined collection)	462
Table 478 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Member of power statistics collection).....	462
Table 479 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Member of pre-defined collection)	463
Table 480 - SMI Referenced Properties/Methods for CIM_StatisticsCollection	463
Table 481 - SMI Referenced Properties/Methods for SNIA_OperationalPowerManifest (Client Defined).....	464
Table 482 - SMI Referenced Properties/Methods for SNIA_OperationalPowerManifest (Provider Support).....	465
Table 483 - SMI Referenced Properties/Methods for SNIA_OperationalPowerManifestCollection (Client De- fined).....	465
Table 484 - SMI Referenced Properties/Methods for SNIA_OperationalPowerManifestCollection (Provider De- fined).....	466
Table 485 - SMI Referenced Properties/Methods for SNIA_OperationalPowerStatisticalData (Optional Ele- ment Stats).....	466
Table 486 - SMI Referenced Properties/Methods for SNIA_OperationalPowerStatisticalData (Top Level System Stats).....	467
Table 487 - SMI Referenced Properties/Methods for SNIA_OperationalPowerStatisticsCapabilities	468
Table 488 - SMI Referenced Properties/Methods for SNIA_OperationalPowerStatisticsService	468

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

INFORMATION TECHNOLOGY – STORAGE MANAGEMENT – Part 3: Common profiles

FOREWORD

- 1) ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.
- 2) The formal decisions or agreements of IEC and ISO on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees and ISO member bodies.
- 3) IEC, ISO and ISO/IEC publications have the form of recommendations for international use and are accepted by IEC National Committees and ISO member bodies in that sense. While all reasonable efforts are made to ensure that the technical content of IEC, ISO and ISO/IEC publications is accurate, IEC or ISO cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees and ISO member bodies undertake to apply IEC, ISO and ISO/IEC publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any ISO, IEC or ISO/IEC publication and the corresponding national or regional publication should be clearly indicated in the latter.
- 5) ISO and IEC do not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. ISO or IEC are not responsible for any services carried out by independent certification bodies.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability shall attach to IEC or ISO or its directors, employees, servants or agents including individual experts and members of their technical committees and IEC National Committees or ISO member bodies for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication of, use of, or reliance upon, this ISO/IEC publication or any other IEC, ISO or ISO/IEC publications.
- 8) Attention is drawn to the normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
- 9) Attention is drawn to the possibility that some of the elements of this ISO/IEC publication may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

International Standard ISO/IEC 24775-3 was prepared by subcommittee 25: Interconnection of information technology equipment, of ISO/IEC joint technical committee 1: Information technology.

This International Standard, together with ISO/IEC 24775-1, ISO/IEC 24775-2 and ISO/IEC 24775-4 to ISO/IEC 24775-8, replaces ISO/IEC 24775, second edition, published in 2011, and constitutes a technical revision.

This edition includes the following significant technical changes with respect to the previous edition:

- a) reorganization into eight parts;
- b) maturity identification using stages; and
- c) new profiles.

The introduction contains a detailed list of the technical changes.

The list of all currently available parts of the ISO/IEC 24775 series, under the general title *Information technology – Storage management*, can be found on the IEC web site.

This International Standard has been approved by vote of the member bodies, and the voting results may be obtained from the address given on the second title page.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

IMPORTANT – The “colour inside” logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this publication using a colour printer.

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

Introduction

The Information Technology – Storage Management is published in several parts. *Information Technology – Storage Management, Part 3 Common Profiles* defines profiles that are used by profiles in other parts of this International Standard. In general, the common profiles do not fully define storage elements, but define non-storage management aspects that are common to storage domains. For example, the Access Points Profile defines a technique that the arrays, switches, or libraries may use to inform clients of non-CIM network interfaces that are available.

Some of the common profiles are based on DMTF profiles. For these profiles, the DMTF profile may be “specialized” to assure SNIA requirements are met.

Parts of this Standard

This International Standard is subdivided into the following parts:

- *Information technology – Storage management – Part 1: Overview*
- *Information technology – Storage management – Part 2: Common architecture*
- *Information technology – Storage management – Part 3: Common profiles*
- *Information technology – Storage management – Part 4: Block devices*
- *Information technology – Storage management – Part 5: Filesystems*
- *Information technology – Storage management – Part 6: Fabric*
- *Information technology – Storage management – Part 7: Host elements*
- *Information technology – Storage management – Part 8: Media libraries*

Changes since the Last Edition

This first multipart edition cancels and replaces the second edition published in 2011 and constitutes a technical revision. This edition includes the following significant changes with respect to the previous edition. The changes fall into three broad categories:

- **Improved organization.** The International Standard has been reorganized into eight parts to provide more information more easily. The parts are:
 - *Part 1 Overview:* The overview book provides a high level overview of the standard.
 - *Part 2 Common Architecture:* This part covers general information about the interface, such as security and protocols.
 - *Part 3 Common Profiles:* This part covers component profiles that extend profiles in other books, such as target ports and job control.
 - *Part 4 Block Devices:* This part covers storage profiles that support various forms of disk storage.
 - *Part 5 Filesystems:* This part covers profiles that support filesystems, such as NAS (Network Attached Storage).
 - *Part 6 Fabric:* This part covers profiles that deal with interconnection of host servers and storage devices, such as switches.
 - *Part 7 Host Elements:* This part covers profiles for storage software on host servers, such as disk partitioning and Host Hardware RAID controllers.
 - *Part 8 Media Libraries:* This part covers profiles that deal with removable media such as tape libraries.
- **Maturity identification.** As material is added to the standard it goes through various stages of maturity. The initial stage is *Experimental*, which is material that has not yet been implemented and is subject to change. The other stages indicate the degree of implementations. The stages are:
 - **Experimental:** Full design review, no commercial implementations.

- **Implemented:** Initial implementations available, may be removed at minor revision.
- **Stable:** Three or more vendors have implemented the identified material, backward compatibility assured, removed only at major revision.
- **Finalized:** Relies solely on Finalized content, deprecated only at major revision.
- **Deprecated:** Obsolete material, may be removed in future revisions.
For a more detailed explanation of each maturity level and its typographical indication, see Clause 4 Typographical Conventions.
- **Expanded scope.** The range of SAN components modeled by the profiles defined in the parts has been greatly expanded.
- New profiles include:
 - *Part 3 Common Profiles:* Serial Attached SCSI (SAS) Target Port, Serial ATA (SATA) Target Ports, SB Target Port, SAS Initiator Ports, ATA Initiator Ports, FC-SB-x Initiator Ports, FCoE Initiator Ports, Power Supply, Fan, Sensors, Base Server, Media Access Device, Storage Enclosure, Software Inventory, Profile Registration, Proxy Server System Management, Operational Power.
 - *Part 4 Block Devices:* Block Storage Views, CKD Block Services, Erasure, Storage Server Asymmetry, Volume Composition, Storage Element Protection, Replication Services, Pools from Volumes, Group Masking and Mapping, Thin Provisioning.
 - *Part 5 Filesystems:* File Export, File Server Manipulation, File Storage, Filesystem, Filesystem Copy Services, Filesystem Performance, Filesystem Quotas, NAS Network Port, Host Filesystem, Filesystem Remote Copy Services.
 - *Part 6 Fabric:* Fibre Channel Security, Fabric Views, Virtual Fabrics, Switch Partitioning, SAS Expander, N Port Virtualizer, Inter Fabric Routing.
 - *Part 7 Host Elements:* Storage HBA, Host Hardware RAID Controller.
 - *Part 8 Media Libraries:* Partitioned Tape Library, Virtual Tape Library, Virtual Tape Library Copy and Library Views.
- The following experimental profiles were removed from the International Standard:
 - *Part 3 Common Profiles:* Security, 3rd Party Authentication, Authorization, Credential Management, Identity Management, Security Role Based Access Control and Security Resource Ownership.
 - *Part 4 Block Devices:* Pool Management Policy.
- The following profiles were deprecated:
 - *Part 3 Common Profiles:* Cascading (replaced by direct use of cascading classes).
 - *Part 4 Block Devices:* Volume Management (and not replaced).
 - *Part 6 Fabric:* Router (and not replaced).
 - *Part 7 Host Elements:* FC HBA (replaced by Storage HBA), SB Multipath Management (and not replaced).
 - *Part 8 Media Libraries:* InterLibraryPort Connection (and not replaced).
- In addition, many of the existing profiles have been enhanced.

This International Standard was prepared by the SNIA (Storage Networking Industry Association)¹. The standard is often referred to as **SMI-S** (*Storage Management Initiative Specification*).

1. Storage Networking Industry Association, 425 Market Street, Suite 1020, San Francisco, CA 94105, U.S.A., <http://www.snia.org>

INFORMATION TECHNOLOGY – STORAGE MANAGEMENT – Part 3: Common profiles

1 Scope

This part of ISO/IEC 24775 defines profiles that are supported by profiles defined in the other parts of the ISO/IEC 24775 series. The first few clauses provide background material that helps explain the purpose and profiles and recipes (a subset of a profile). Common port profiles are grouped together since they serve as transport-specific variations of a common model. The port profiles are followed by other common profiles. Clause 50 presents recipes that span multiple profiles.

2 Normative references

The following documents, in whole or in part, are normatively referenced in the document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document, including any amendments, applies.

ISO/IEC 14776-413, *Information technology – Small computer system interface (SCSI) – Part 413: Architecture Model - 3 (SAM-3)*
(herein referred to as SAM)

ISO/IEC 14776-453, *Information technology – Small computer system interface (SCSI) – Part 453: Primary Commands - 3*
(herein referred to as SPC)

ISO/IEC 24775-2, *Information technology – Storage Management – Part 2: Common architecture*

DMTF DSP0207, *WBEM URI Mapping 1.0*
http://www.dmtf.org/standards/published_documents/DSP0207.pdf

DMTF DSP1004, *DMTF Base Server Profile 1.0.1*

DMTF DSP1004, *Base Server Profile 1.0.1*
http://www.dmtf.org/standards/published_documents/DSP1004_1.0.1.pdf

DMTF DSP1009:2006, *Sensors Profile 1.0.0*
http://www.dmtf.org/standards/published_documents/DSP1009.pdf

DMTF DSP1011:2006, *Physical Asset Profile 1.0.2*
http://www.dmtf.org/standards/published_documents/DSP1011_1.0.2.pdf

DMTF DSP1013:2006, *Fan Profile 1.0.1*
http://www.dmtf.org/standards/published_documents/DSP1013_1.0.1.pdf

DMTF DSP1015:2006, *Power Supply Profile 1.0.1*
http://www.dmtf.org/standards/published_documents/DSP1015_1.0.1.pdf

DMTF DSP1019:2009, *Device Tray Profile 1.0.0*
http://www.dmtf.org/standards/published_documents/DSP1019_1.0.0.pdf

DMTF DSP1022:2010, *CPU Profile 1.0.1*
http://www.dmtf.org/standards/published_documents/DSP1022_1.0.1.pdf

DMTF DSP1023:2009, *Software Update Profile 1.0.1*
http://www.dmtf.org/standards/published_documents/DSP1023_1.0.1.pdf

DMTF DSP1025:2009, *Software Update Profile 1.0.0*

http://www.dmtf.org/standards/published_documents/DSP1025_1.0.0.pdf

DMTF DSP1033:2007, *Profile Registration Profile 1.0.0*

http://www.dmtf.org/standards/published_documents/DSP1033_1.0.0.pdf

DMTF DSP1052, *Computer System Profile 1.0.0*

http://www.dmtf.org/standards/published_documents/DSP1052_1.0.0.pdf

DMTF DSP1102:2009, *Launch in Context Profile 1.0.0b*

http://www.dmtf.org/standards/published_documents/DSP1102_1.0.0b.pdf

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

3 Terms, definitions and abbreviations

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 24775-2:2014 apply.

3.2 Abbreviations

FC-SB-X

Fibre Channel Single-Byte command set used in FICON™¹ devices

SAS

Serial Attached SCSI

SATA

Serial ATA

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

1. FICON™ is an example of a suitable product available commercially. This information is given for the convenience of users of this International Standard and does not constitute an endorsement of this product by SNIA or any standards organization.

4 Profile introduction

4.1 Profile overview

4.1.1 General

A profile is a specification that defines the CIM model and associated behavior for an autonomous and self-contained management domain. The CIM model includes the CIM Classes, Associations, Indications, Methods and Properties. The management domain is a set of related management tasks. A profile is uniquely identified by the name, organization and version.

In SMI-S, a profile describes the management interfaces for a class of storage subsystem, typically realized as a hardware or software product. For example, SMI-S includes profiles for arrays, FC-Switches, and logical volume manager software. The boundaries chosen for SMI-S profiles are often those of storage products, but some vendors may package things differently. For example, one vendor may choose to package an Array and an FC Switch into a single product; this can be handled in SMI-S by implementing the Array and FC Switch Profiles for this product.

A profile may add restrictions to usage and behavior, but cannot change CIM defined characteristics. For example, if a property is required in the CIM model, then it is required in a profile. On the other hand, a profile may specify that a property is required even if it is not required by the general CIM model.

In SMI-S, profiles serve several purposes:

- Organization - the SMI-S object model (see *Information Technology – Storage Management, Part 2 Common Architecture 7 Object Model General Information*) is presented as a set of profiles, each describing a type of storage element or behavior,
- Certification - SMI-S profiles form the basis for CTP certification,
- Discovery- profiles are registered with the CIM Server and advertised to clients as part of the CIM model and using SLP (see *Information Technology – Storage Management, Part 2 Common Architecture 10 Service Discovery*). An SMI-S client uses SLP to determine which CIM Servers host profiles it wishes to manage, then uses the CIM model to discover the actual configurations and capabilities.

A subprofile is a profile that specifies a subset of a management domain. A subprofiles's CIM elements are scoped within a containing profile. Multiple profiles may use the same subprofile. A subprofile is uniquely identified by the name, organization and version.

A profile specification may include a list of the subprofiles it uses. The included subprofiles may be optional or mandatory by the scoping profile. The behavior of a profile is specified in this profile and its included subprofiles.

For example, target devices such as RAID arrays and tape libraries may support Fibre Channel or parallel SCSI connectivity. SMI-S includes an FC Target Port Subprofile and a Parallel SCSI Target Port Subprofile that may optionally be supported by profiles representing target devices. The elements defined in the port subprofiles are scoped to the ComputerSystem in the profile. For example, each LogicalPort subclass has a SystemDevice association to the profile's ComputerSystem.

In addition to sharing the purposes of profiles (as described in this section), subprofiles have these purposes:

- Optional behavior - a profile may allow, but not require, an implementation to support a subprofile. Although a subprofile does not describe a full product, a subprofile should describe an aspect of a product that is recognizable to an knowledgeable end-user such as a storage administrator,
- Reuse of functionality - some storage management behavior is common across different types of storage elements. For example, block virtualization is managed similarly in RAID arrays and

logical volume managers. These common sets of functionality are specified as profiles that are shared by several other profiles.

- **Decomposition** - certain functionality may not be reused multiple places, but is complicated enough to document as a separate profile. For example, Disk Partition management is only used in the Host Discovered Resources profiles, but is complicated enough that it has been documented as a separate profile.

4.1.2 Terminology

A profile collects included subprofiles and provides the filler needed to define the management interfaces of a particular type of subsystem. Profiles are separated into two groups. *Storage profiles* define the management interfaces for storage subsystems such as arrays or FC switches. *Generic profiles* define management interfaces for generic systems that are related to storage management. Storage and generic profiles are specified the same way in SMI-S, but generic profiles are not certified as free-standing entities, only as a dependency of a storage profile.

A *Package* is a profile that whose implementation is mandatory to comply with the requirements of all of its containing top-level profiles. Since a package is always mandatory, it is not registered with the CIM Server. Packages provide decomposition in the International Standard.

Profiles may be related by *specialization* - where several profiles (or subprofiles) share many common elements, but are specialized for specific implementations. The SMI-S Security profiles are an example; the specializations (Authorization Profile, Security Resource Ownership Profile,...) share some classes and behavior. Profile specialization is only an artifact of the International Standard. It saves the reader from reading common aspects in multiple places and help the International Standard stay consistent across the specialized profiles. There is no information in the CIM model about the relationship between generic and specialized profiles.

4.2 Format for Profile Specifications

For each profile there is a set of information that is provided to specify the characteristics and requirements of the profile. Subprofiles are also defined using this format, but they are clearly identified as subprofiles.

Each profile or subprofile is defined in subsections that are described in Table 1.

NOTE CIM schema diagrams are logically part of a profile description. However, they can be rather involved and cannot be easily depicted in a single diagram. As a result, the reader is advised to refer to DMTF characterizations of CIM schema diagrams.

Table 1 - Profile Components<RegularWeight> (part 1 of 2)

Profile Element	Goal
Description	This section provides a description of the profile and model including an overview of the objectives and functionality. <i>Functionality</i> is described in a bullet-form in this section that includes functionality provided by the subprofiles referenced by the profile. If a function is provided by a subprofile, this is indicated, including whether the subprofile is optional or required. Functionality listed in the profile is organized by Levels, and within each Level by FCAPS category, as defined in the SMI-S functionality matrix section <link>. <i>Instance Diagrams</i>: One or more instance diagrams to highlight common implementations that employ this section of the Object Model. Instance diagrams also contain classes and associations but represent a particular configuration; multiple instances of an object may be depicted in an instance diagram. Finally, this section may include supporting text for recipes, properties, and methods as needed.
Health & Fault Management Considerations	If a profile provides optional Health & Fault Management capabilities, then this section describes the specifics of these capabilities, including: • A table of the classes that report health information. • Tables of possible states of the OperationalStatus and HealthState attributes and descriptions for those elements that report state. • Cause and Effect associations. • Standard Errors produced (including Alert Indications, Errors, CIM Errors, and Health Related Live Cycle Events).
Cascading Considerations	A Profile may be a cascading profile. A cascading profile is any Profile that supports the Cascading Subprofile as either a mandatory or recommended subprofile. If the profile is a cascading profile, this section documents cascading considerations in each of the following areas: • Cascaded Resources – Defines the type of resources in the Cascading Profile that are associated to what type of resources in the Leaf Profile and the association. • Ownership Privileges – Identifies the Resource Control Privileges (on leaf resources) that are established by the Cascading Profile. • Limitations on Cascading Subprofile – Identifies any limitations on the Cascading Subprofile that are imposed by the Cascading in effect
Supported Profiles, Subprofiles, and Packages	A list of the names and versions of subprofiles and packages supported by a profile.
Methods of the Profile	This section documents the methods used in this profile. All methods used in recipes shall be documented; optional methods (those not used in recipes) may also be included.

Table 1 - Profile Components<RegularWeight> (part 2 of 2)

Profile Element	Goal
Client Considerations and Recipes	This section documents a set of "recipes" that describe the CIM operations and other steps required to accomplish particular tasks. These recipes do not define the upper bound of what a CIM Server may support, however, they define a lower bound. That is, a CIM Provider implementation shall support these recipes as prescribed to be SMI-S compliant. NOTE A recipe that is defined as part of a subprofile is only required if the subprofile is implemented. All optional behavior in a profile shall be described in a recipe and shall have a capabilities property a client can test to determine whether the optional behavior is supported. The actual capabilities properties are documented in "Classes Used in the Profile" in this table.
CIM Elements	A table listing the classes, associations, subprofile, packages, and indication filters that this profile (or subprofile) supports, and a brief description of each. Everything listed in this section is mandatory for the profile or subprofile. This section shall not list optional elements. Prior to SMI-S 1.1.0, CIM did not have standard language for indication filters; SMI-S 1.0.x used the proposed WQL query language. This version of SMI-S uses the CQL standard query language. WQL is also supported for backwards compatibility. The Description column for an indication filter specifies whether the filter string is compliant to CQL or WQL. If neither is stated, then the string complies to both CQL and WQL.
Classes Used in the Profile	This section provides one table per class and lists each required and recommended property. For each required or recommended property a brief description on what information is to be encoded is identified. The class tables include a "Flags" column. This can contain "C" (the property is a correlatable name or a format for a name), "D" (the property is a durable name), "M" (the property is modifiable), or "N" (null is a valid value).

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

5 Typographical Conventions

5.1 Maturity Model

In addition to informative and normative content, this International Standard includes guidance about the maturity of emerging material that has completed a rigorous design review but has limited implementation in commercial products. This material is clearly delineated as described in Clause 5. The typographical convention is intended to provide a sense of the maturity of the affected material, without altering its normative content. By recognizing the relative maturity of different sections of the International Standard, an implementer should be able to make more informed decisions about the adoption and deployment of different portions of the International Standard in a commercial product.

This International Standard has been structured to convey both the formal requirements and assumptions of the Information Technology – Storage Management API and its emerging implementation and deployment lifecycle. Over time, the intent is that all content in the International Standard will represent a mature and stable design, be verified by extensive implementation experience, assure consistent support for backward compatibility, and rely solely on content material that has reached a similar level of maturity. Unless explicitly labeled with one of the subordinate maturity levels defined for this International Standard, content is assumed to satisfy these requirements and is referred to as “Finalized”. Since much of the evolving International Standard content in any given release will not have matured to that level, this International Standard defines three subordinate levels of implementation maturity that identify important aspects of the content’s increasing maturity and stability. Each subordinate maturity level is defined by its level of implementation experience, its stability and its reliance on other emerging standards. Each subordinate maturity level is identified by a unique typographical tagging convention that clearly distinguishes content at one maturity model from content at another level.

5.2 Experimental Maturity Level

No material is included in this International Standard unless its initial architecture has been completed and reviewed. Some content included in this International Standard has complete and reviewed design, but lacks implementation experience and the maturity gained through implementation experience. This content is included in order to gain wider review and to gain implementation experience. This material is referred to as “Experimental”. It is presented here as an aid to implementers who are interested in likely future developments within the SMI International Standards. The contents of an Experimental profile may change as implementation experience is gained. There is a high likelihood that the changed content will be included in an upcoming revision of the International Standard. Experimental material can advance to a higher maturity level as soon as implementations are available. Figure 1 is a sample of the typographical convention for Experimental content.

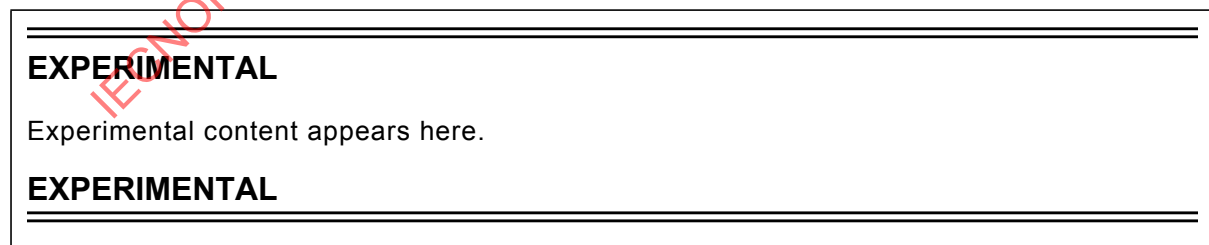


Figure 1 - Experimental Maturity Level Tag

5.3 Implemented Maturity Level

Profiles for which initial implementations have been completed are classified as “Implemented”. This indicates that at least two different vendors have implemented the profile, including at least one provider implementation. At this maturity level, the underlying architecture and modeling are stable, and changes in future revisions will be limited to the correction of deficiencies identified through additional implementation experience. Should the material become obsolete in the future, it shall be deprecated in a minor revision of the

International Standard prior to its removal from subsequent releases. Figure 2 is a sample of the typographical convention for Implemented content.

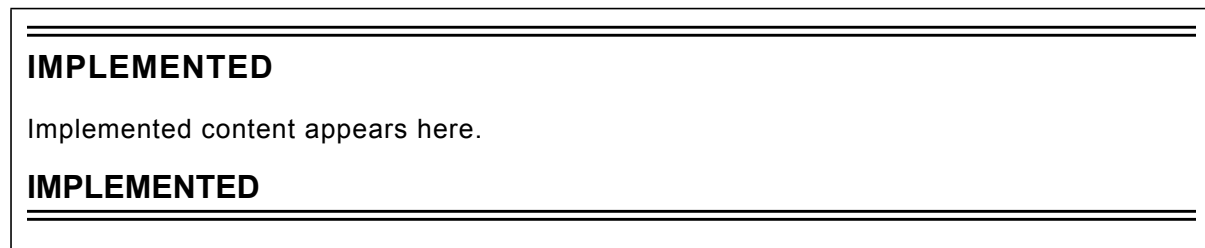


Figure 2 - Implemented Maturity Level Tag

5.4 Stable Maturity Level

Once content at the Implemented maturity level has garnered additional implementation experience, it can be tagged at the Stable maturity level. Material at this maturity level has been implemented by three different vendors, including both a provider and a client. Should material that has reached this maturity level become obsolete, it may only be deprecated as part of a minor revision to the International Standard. Material at this maturity level that has been deprecated may only be removed from the International Standard as part of a major revision. A profile that has reached this maturity level is guaranteed to preserve backward compatibility from one minor International Standard revision to the next. As a result, Profiles at or above the Stable maturity level shall not rely on any content that is Experimental. Figure 3 is a sample of the typographical convention for Implemented content.

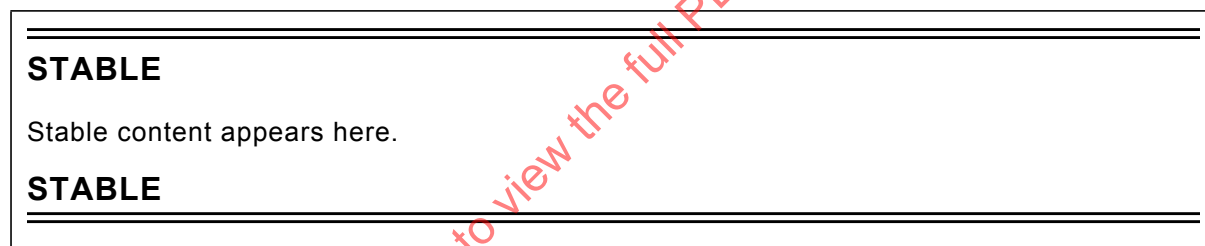


Figure 3 - Stable Maturity Level Tag

5.5 Finalized Maturity Level

Content that has reached the highest maturity level is referred to as "Finalized." In addition to satisfying the requirements for the Stable maturity level, content at the Finalized maturity level shall solely depend upon or refine material that has also reached the Finalized level. If International Standard content depends upon material that is not under the control of the SNIA, and therefore not subject to its maturity level definitions, then the external content is evaluated by the SNIA to assure that it has achieved a comparable level of completion, stability, and implementation experience. Should material that has reached this maturity level become obsolete, it may only be deprecated as part of a major revision to the International Standard. A profile that has reached this maturity level is guaranteed to preserve backward compatibility from one minor International Standard revision to the next. Over time, it is hoped that all International Standard content will attain this maturity level. Accordingly, there is no special typographical convention, as there is with the other, subordinate maturity levels. Unless content in the International Standard is marked with one of the typographical conventions defined for the subordinate maturity levels, it should be assumed to have reached the Finalized maturity level.

5.6 Deprecated Material

Non-Experimental material can be deprecated in a subsequent revision of the International Standard. Sections identified as "Deprecated" contain material that is obsolete and not recommended for use in new development efforts. Existing and new implementations may still use this material, but shall move to the newer approach as soon as possible. The maturity level

of the material being deprecated determines how long it will continue to appear in the International Standard. Implemented content shall be retained at least until the next revision of the specialization, while Stable and Finalized material shall be retained until the next major revision of the International Standard. Providers shall implement the deprecated elements as long as it appears in the International Standard in order to achieve backward compatibility. Clients may rely on deprecated elements, but are encouraged to use non-deprecated alternatives when possible.

Deprecated sections are documented with a reference to the last published version to include the deprecated section as normative material and to the section in the current International Standard with the replacement. Figure 4 contains a sample of the typographical convention for deprecated content.



Figure 4 - Deprecated Tag

6 Recipe overview

6.1 Recipe concepts

Recipe: A set of instructions for making something from mixing various ingredients in a particular sequence. The set of ingredients used by a particular recipe is scoped by the particular profile, subprofile or some other well-defined context in which that recipe is defined.

A recipe shall specify an interoperable means for accomplishing a particular task across all conformant implementations. However, a recipe does not necessarily specify the only set of instructions for accomplishing that task. Nor are all tasks that may be accomplished necessarily specified by the set of recipes defined for a particular profile or subprofile.

In order to compress the document, some recipes are implied or assumed. This would include, for instance, that the set of available, interoperable properties are those explicitly defined by a particular profile or subprofile. In general, any CIM intrinsic read methods on profile or subprofile models are implied. However, CIM intrinsic write methods (Create/Delete/Modify) should not be assumed unless explicitly listed in the profile or subprofile definition with a well defined semantic.

For a profile or subprofile, the set of all defined and implied recipes defines the range of behavior across for which interoperability is mandatory for all conformant implementations. Unless specifically defined in a recipe, other sequences of actions (even simple Create/Delete instance requests) are not guaranteed to have the same results across multiple implementations.

Each recipe defines an interoperable series of interactions (between a Information Technology – Storage Management Client and a Information Technology – Storage Management Server) required to manage storage devices or applications. Another goal is to list the operations required for the CIM Client realize functionality. It is not a goal to comprehensively express the programming logic required to implement the recipe in any particular language. In fact, recipes are limited to the expression of CIM or SLP operations, and may simply reference or describe any of the implementation that may be required beyond that.

6.2 Recipe Pseudo Code conventions

6.2.1 Overview

A recipe's instructions are written using the pseudo code language defined in this section.

All recipes are prefixed with a summary narrative of the functionality being implemented. This summary may be included explicitly as part of the recipe or reference to the appropriate narrative that can be found elsewhere in the International Standard.

NOTE The use of optional features (profiles or subprofiles) in recipes shall be clearly identified.

CIM Operations and their parameters are taken directly from the *CIM Operations Over HTTP* specification. It is assumed that these methods are being called on the CIM Client API. Arrays grow in size automatically.

6.2.2 General Syntax

<condition>	logical statement that evaluates to true (Boolean)
!<condition>	tests for false (Boolean)
<action>	unspecified list of programming logic that is not important to the understanding of the reader for a particular recipe.
<EXIT: <i>success message</i>>	Exits the recipe with a success status code. The condition that resulted in the call to exit the recipe was allowable. The implementation subjected to the recipe behaves in accordance to this International Standard.

<ERROR! error condition> Exits the recipe with a failure status code. The condition that resulted in the call to the exit the recipe was not allowable. The implementation subjected to the recipe does not behave in accordance with this International Standard.

@{recipe} logic flow is contained within the specification of the recipe elsewhere in the specification

<variable> some variable

6.2.3 CIM related variable and methods

6.2.3.1 CIM Instances and Object Names

\$name represents a single instance (CIMInstance) with a given variable name

\$name.property represents a property in a single instance (CIMInstance)

\$name.getObjectPath()
method returns a object name, REF, to the CIM Instance

\$name.getNameSpace()
method returns the namespace name for the CIM Instance or Object Name

{value1, value2 ...}
an anonymous array, comprised of selected values of a given type; an anonymous array is an array that is not referable by a variable

EXAMPLE

{ "Joe", "Fred", "Bob", "Celma" }

\$name[] represents an array of instances (CIMInstances) with a given variable name; array are initialized by constructing an anonymous array.

EXAMPLE

Names = { "Joe", "Fred", "Bob", "Celma" }

\$name-> represents an object path name (CIMObjectPath)

\$name->[] represents an array of object names of a given name

\$name->property
represents a property of object \$name

\$name[].size() returns the number of CIM instances in the array

\$name->[].length returns the number of CIM object names in the array

#name[].length returns the number of variable elements in the array

%name[].length returns the number of method arguments elements in the array

6.2.3.2 Extrinsic method arguments

%name represents a CIM Argument that can contain any CIM or other variable.

%name[] represents an array of CIM Arguments

6.2.3.3 Other Variables

#name neither CIM Instance nor Object Name variable. The type may be a string, number or some other special type. Types are defined in the CIM Specification 2.2.

#name[]	a non-CIM variable array
"literal"	some string literal

6.2.4 Data structure

Variables can be collected by an array. The array can be indexed by other variable (see 6.2.3.3).

Arguments are always indexed by strings. In other words, the arguments are retrieved from the array by name.

6.2.5 Operations

=	assigns right value to left value
==	test for equivalency
!=	test for not equivalency
<	true if the left argument is numerically less than the right argument.
>	true if the left argument is numerically greater than the right argument.
<=	true if the left argument is numerically less than or equal to the right argument.
>=	true if the left argument is numerically greater than or equal to the right argument.
&&	condition A AND condition B
 	condition A OR condition B
+, -, *, /	addition, subtraction, multiplication and division, respectively
++, --	increment and decrement a variable, respectively; placement of the operator relative to the variable determines whether the operation is completed before or after evaluation

EXAMPLE

```
#i = 1
#names[] = {"A", "B", "C"}
"B" == #names[++#i] is true
2 == #i is true
```

EXAMPLE

```
#i = 2
#names[] = {"A", "B", "C"}
"B" == #names[#i++] is true
3 == #i is true
// comments
```

nameof returns an Object Name given a CIM Instance. This unitary operator does nothing in other usages.

ISA tests for the name of the CIM Instance or object name

```
EXAMPLE if ($SomeName-> ISA CIM_StorageVolume) {
<The Object Name is a reference to a CIM_StorageVolume >
}
```

6.2.6 Control operations

The pseudocode used in this International Standard relies on control operators common to most high-level languages. For example:

- **for**

EXAMPLE

```
for #x in <variable array> {
  <actions>
}
```

- **if**

EXAMPLE

```
if (<condition>) {
  <actions>
};
if (<condition>) {
  <actions>
} else {
  <alternate actions>
}
```

- **do/while**

EXAMPLE

```
do {
  <actions>
} while (<condition>)
```

- **continue**

Within a **for** loop: initialize loop variable to next available value and restart loop body. Terminate loop if no more loop variable values available. Within a **do/while** loop: transfer control immediately to **while** test.

EXAMPLE

```
for #i in <array> {
  if (<some condition>)
    continue; // process next loop variable
  <alternative>
}
```

- **break**: interrupts the sequence of statement execution within a loop block and exits the loop block altogether. The looping condition is not re-evaluated. Statement execution starts at the next statement outside of the loop block.

- **exit**

Terminate recipe instantly, including termination of any callers.

EXAMPLE

```
if (<unexpected condition>)
  exit
```

6.2.7 Functions

6.2.7.1 Function declaration

A function definition is of the form *sub functionName()*, followed by the body of the function enclosed in braces. If parameters are to be passed to a function, then are expressed as a comma-separated list of arguments within the parentheses following the function name. Each argument is comprised of a data type and an accompanying argument name.

Functions are declared at the beginning of a recipe.

```
sub functionName(integer nArg1, Class &cArg2) {
```

```
<actions>
}
```

6.2.7.2 Function invocation

A function invocation is of the form `&functionName()`. If parameters are to be passed to a function, then are expressed as a comma-separated list within the parentheses following the function name.

```
&functionName(5, pClass)
```

6.2.8 Exception handling

All operations may produce exceptions or errors. The following construct is used to test for particular errors. Once a particular error is caught, then special exception handling logic is processed. Only CIM Errors can be caught.

```
try {
    <actions>
}
    catch (CIM Exception $Exception) {
        <recovery actions>
    }
    The error received may also be thrown
    throw $Exception
```

The error response returned from the SMI-S implementation is treated as a exception, a "CIM Exception". The catch condition is expressed in terms of the CIM status code returned (e.g., CIM_ERR_NOT_FOUND) as defined in the CIM Operations specification.

The `$Exception` variable contains a Error instance. The `$Exception` CIM Instance may be examined like any other CIM Instance. In this language, the `$Exception` is never null even if the SMI-S implementation does provide one. In this case, the `$Exception` CIM Instance is empty with the exception of the `CIMStatusCode` and `CIMStatusCodeDescription` properties. This properties are populated with the Status and Description returned in the error response from the SMI-S implementation.

6.2.9 Built-in functions

a) `boolean = compare(<variable>, <variable>)`

- 1) Used to determine if two variables of the same type are equivalent
- 2) The variables shall not be CIM instances or object names nor other complex data types or structures
- 3) The variables shall be of the same type

b) `$instance = newInstance("CIM Classname")`

- 1) Creates a CIM instance, which does not exist in the CIMOM (yet), that can be later filled in with properties and passed to `CreateInstance`. The namespace is assumed to be the same that the CIM client connected to.

c) `$instance - newInstance("CIM Namespace", "CIM Classname")`

- 1) Variable of the above method that has the namespace name as an argument

d) `boolean = contains(<test value>, <variable array>)`

- 1) Used to test if the variable array contains a value equivalent to the test variable
- 2) The array shall be of variables of the same types as the test variable.

- 3) If the equivalency is found with at least one value then the function returns true, else false is returned.
 - 4) If the array is not a simple, or non-CIM, data type, then the test value shall be a CIM property, \$SomeInstance.SomeProperty or \$SomeObjectName->SomeProperty
- e) %Argument = newArgument("Argument Name", <variable>)
- 1) Creates a CIM Argument of a given name containing a value, CIM or non-CIM
- f) \$ObjectPath-> = newObjectPath("Class name", "NameSpace name")
- 1) Returns a new ObjectPath, built from the supplied arguments;
 - 2) Required to perform the EnumerateInstances and EnumerateInstanceNames operations
- g) #stringArray[] = #stringVariable.split(#stringParam or "string literal")
- 1) Returns an array of strings, built by splitting the string variable around matches of the supplied string parameter
 - 2) Divides the string into substrings, using the string parameter as a delimiter, returning the substrings in an array in the order in which they occurred in the string variable. If there are no occurrences of the string parameter, then the array returned contains only one string element equal to the original string variable.
- h) #intValue = Integer(#stringVariable)
- 1) Returns the integer that the supplied string represents. If the supplied string does not represent an integer, then an error is thrown.
 - 2) The function will parse and return signed or unsigned integers up to 64-bits in size, and will accept the hyphen '-' character in the 8-bit ASCII-range of UTF-8 as the first character in the string to indicate a negative number.
- i) #datetimeVariable = Datetime(#stringVariable)
- 1) Returns a variable of Datetime type, as defined by section 2.2.1 the CIM Infrastructure Specification v1.3, that the supplied string represents. If the supplied string does not represent a DateTime object, then an error is thrown.
 - 2) This function will accept strings of the format described in the CIM Infrastructure Specification, including both timestamps and intervals, zero-padded to 25-characters, and will recognize Datetime strings containing asterisk ("*") characters for fields that are not significant.

6.2.10 Extrinsic method calls

```
<variable> = InvokeMethod ($someobjectname->, "Method Name",
    %InArguments[], %OutArguments[])
```

EXPERIMENTAL

7 Generic Target Ports Profile

7.1 Synopsis

Profile Name: Generic Target Ports (Component Profile)

Version: 1.4.0

Organization: SNIA

CIM Schema Version: 2.9.0 (specialized profiles may need later versions)

Related Profiles for Generic Target Ports: Not defined in this International Standard.

Central Class: CIM_LogicalPort

Scoping Class: a CIM_System in a separate autonomous profile

The Generic Target Port Profile models the generic behavior of target ports in storage systems such as disk arrays and tape libraries.

This abstract profile specification shall not be directly implemented; implementations shall be based on a profile specification that specializes the requirements of this profile.

7.2 Description

The Generic Target Port Profile models the generic behavior of target ports in storage systems such as disk arrays and tape libraries. Separate profiles specialize the Generic Target Port Profile for Fibre Channel, iSCSI, and other transports. The primary classes of the Generic Target Port Profile are LogicalPort and ProtocolEndpoint, as shown in Figure 5. Instances of subclasses of a LogicalPort (e.g., FCPort, EthernetPort) represent the logical aspects of ports, independent from command protocols (such as SCSI). Instances of subclasses of ProtocolEndpoint (e.g., SCSIProtocolEndpoint or ATA ProtocolEndpoint) represent command protocols in use on the port.

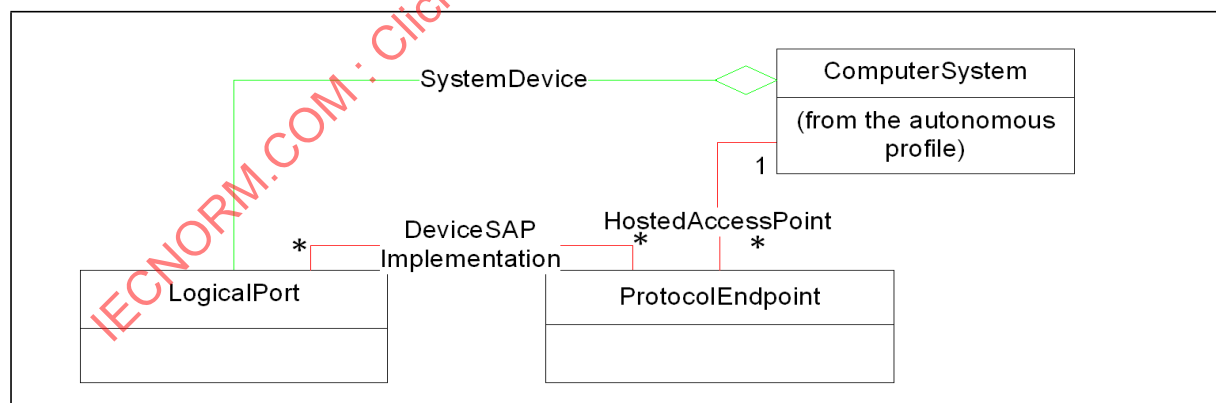


Figure 5 - Generic Target Port Classes

7.3 Implementation

7.3.1 General

Subclasses of ProtocolEndpoint represent command protocols supported by the port. SCSIProtocolEndpoint represents SCSI as a protocol, independent of specific transports or device types – i.e., the behavior described in the SCSI Primary Commands (SPC) and SCSI Architecture Model (SAM) specifications from T10. SCSIProtocolEndpoint.Role indicates whether this protocol endpoint instance represents a SCSI Target or target. For target port

profiles, Role shall be Target” or “Both Initiator and Target”. iSCSIProtocolEndpoint specializes SCSIProtocolEndpoint with additional iSCSI-specific properties.

ATAProtocolEndpoint represents the ATA command protocol. SBPProtocolEndpoint represents Single Byte protocol used with mainframes. ProtocolEndpoint is associated to a System instance with Hosted Access Point.

LogicalPort subclasses specify the type of transport. If the port is subclassed directly from LogicalPort it indicates it is connected to a bus. If the port is further subclassed from NetworkPort it indicates the port is capable of being used in a network. Specializations of this profile shall specify the appropriate subclass of LogicalPort. Figure 6 shows the subclasses of LogicalPort.

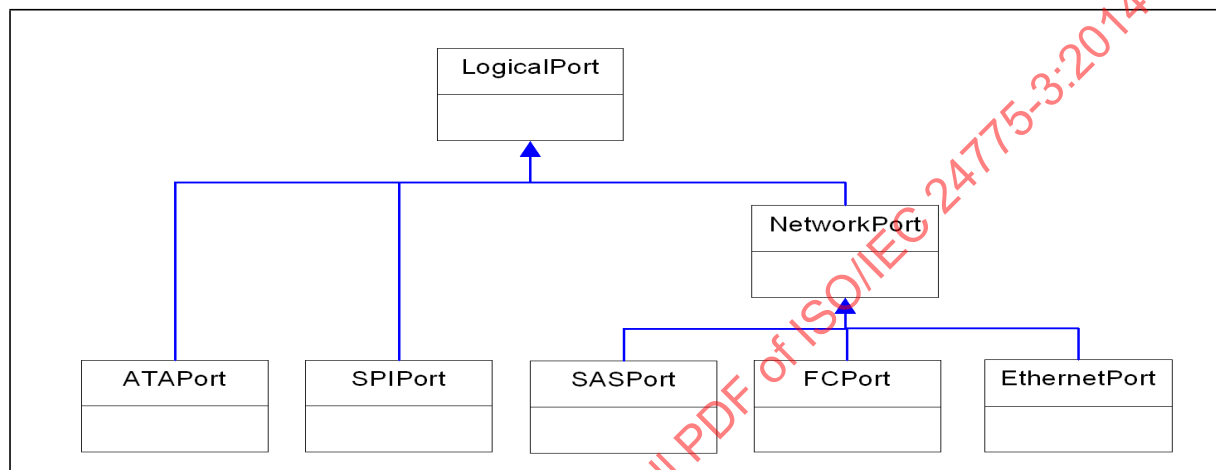


Figure 6 - LogicalPort Class Hierarchy

A property on LogicalPort called UsageRestriction indicates whether the port is restricted to use as a “front end” (target) or a “back end” (Target) interface or both. Note that port may not have a restriction and the actual point-in-time role is modeled in SCISProtocolEndpoint.Role. SystemDevice associates LogicalPort to a System.

ProtocolEndpoint and LogicalPort are associated with DeviceSAPImplementation. For most transports, the command protocol is implemented in the port hardware and there is 1-1 cardinality between the LogicalPort and ProtocolEndpoint instances. iSCSI is an exception, many-to-many relationships are possible between EthernetPort and iSCSIProtocolEndpoint instances.

ProtocolController (in the Mapping and Masking Profile) represents the SCSI (or SB) ‘view’ of ports and logical devices seen by target systems (e.g., arrays). In a system supporting Mapping and Masking, zero or more views exist; defined by the customer to expose subsets of logical units to certain Targets. SAPAvailableForElement connects ProtocolEndpoint from a target ports profile to SCSIProtocolController instances from the Mapping/Masking Profile. iSCSI and SB have protocol-specific, secondary uses of ProtocolController.

Figure 7 depicts a generic storage device with elements from a target ports profile, the Mapping/Masking Profile, and a target device profile. The LogicalDevice object represents logical units that are visible to external systems. It is subclassed to StorageVolume, TapeDrive, etc. to identify the device type.

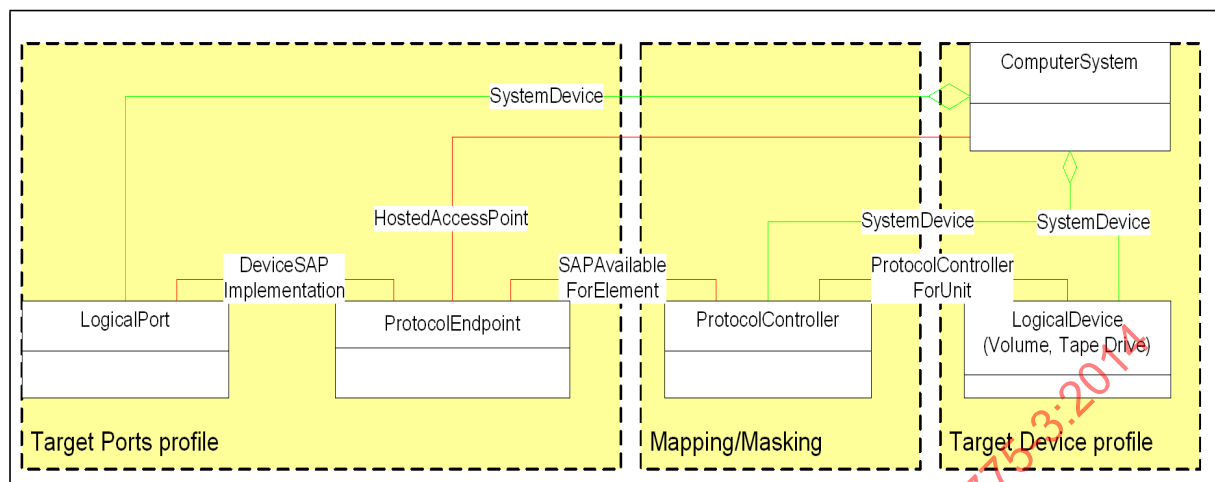


Figure 7 - Generic Target with LUN Masking

7.3.2 Modeling SCSI/SB Logical Units

The SCSI standard inquiry response includes a Device Type property with integers representing types of devices. Most of these devices types have a CIM analog. Devices that are used primarily for management are modeled as SCSIArbitraryLogicalUnit. SCSIArbitraryLogicalUnit.DeviceType maps to SCSI device types. Table 2 describes how common storage devices are modeled in CIM.

Table 2 - Modeling of Common Storage Devices in CIM

SCSI Device Type	Inquiry Device Type	LogicalDevice subclass
DirectAccessDevice	0	DiskDrive or StorageVolume
SequentialAccessDevice	1	TapeDrive
WriteOnceDevice	4	WormDrive
CD-ROM	5	CDROMDrive
MediaChanger	8	MediaTransferDevice
ArrayController	0xc	SCSIArbitraryLogicalUnit DeviceType="SCSI SCC Device"
SES	0xd	SCSIArbitraryLogicalUnit DeviceType="SCSI SES"
Other		SCSIArbitraryLogicalUnit DeviceType="Other"
Unknown		SCSIArbitraryLogicalUnit DeviceType="Unknown"
DirectAccessDevice	0	DiskDrive or StorageVolume

All devices (logical units) visible to external systems shall be modeled.

7.4 Methods of the Profile

7.4.1 Extrinsic methods

None

7.4.2 Intrinsic methods

The profile supports read methods and association traversal. Specifically, the list of intrinsic operations supported are as follows:

- GetInstance
- Associators
- AssociatorNames
- References
- ReferenceNames
- EnumerateInstances
- EnumerateInstanceNames

7.5 Use Cases

Not defined in this International Standard.

7.6 CIM Elements

7.6.1 General

Table 3 describes the CIM elements for Generic Target Ports.

Table 3 - CIM Elements for Generic Target Ports

Element Name	Requirement	Description
7.6.2 CIM_DeviceSAPImplementation	Mandatory	Associates front-end LogicalPort and target ProtocolEndpoint.
7.6.3 CIM_HostedAccessPoint	Mandatory	Associates ComputerSystem to ProtocolEndpoint.
7.6.4 CIM_LogicalPort	Mandatory	Represents the logical aspects of the physical port and may have multiple associated protocols.
7.6.5 CIM_ProtocolEndpoint	Mandatory	ProtocolEndpoint representing support for SCSI, ATA, or SB command set.
7.6.6 CIM_SystemDevice (Port)	Mandatory	Associates ComputerSystem to LogicalPort.

7.6.2 CIM_DeviceSAPImplementation

Associates front-end LogicalPort and target ProtocolEndpoint.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 4 describes class CIM_DeviceSAPImplementation.

Table 4 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Requirement	Description & Notes
Dependent	Mandatory	Reference to SCSI, ATA, or SB ProtocolEndpoint.
Antecedent	Mandatory	Reference to Port.

7.6.3 CIM_HostedAccessPoint

Associates ComputerSystem to ProtocolEndpoint. Limit to targets.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 5 describes class CIM_HostedAccessPoint.

Table 5 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in the referencing profile.
Dependent	Mandatory	Reference to SCSIProtocolEndpoint, ATAProtocolEndpoint or SBProtocolEndpoint.

7.6.4 CIM_LogicalPort

Represents the logical aspects of the physical port and may have multiple associated protocols.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 6 describes class CIM_LogicalPort.

Table 6 - SMI Referenced Properties/Methods for CIM_LogicalPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus	Mandatory	
UsageRestriction	Mandatory	Shall be 2 for ports restricted to Front-end only or 4 if the port is unrestricted.
PortType	Mandatory	VALUE and DESC should be set appropriately for each specialized target port profile.

7.6.5 CIM_ProtocolEndpoint

ProtocolEndpoint representing support for SCSI, ATA, or SB command set. Shall be subclassed in specialized profiles.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 7 describes class CIM_ProtocolEndpoint.

Table 7 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType	Mandatory	Shall be 1 (Other).
OtherTypeDescription	Mandatory	Shall be the string 'SCSI', 'ATA', or 'SB'.

7.6.6 CIM_SystemDevice (Port)

Associates ComputerSystem to LogicalPort.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 8 describes class CIM_SystemDevice (Port).

Table 8 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to ComputerSystem in the referencing profile.
PartComponent	Mandatory	Reference to Port.

EXPERIMENTAL

EXPERIMENTAL

8 Parallel SCSI (SPI) Target Ports Profile

8.1 Synopsis

Profile Name: SPI Target Ports (Component Profile)

Version: 1.4.0

Organization: SNIA

CIM Schema Version: 2.9.0

Table 9 describes the related profiles for SPI Target Ports.

Table 9 - Related Profiles for SPI Target Ports

Profile Name	Organization	Version	Requirement	Description
Indication	SNIA	1.5.0	Mandatory	

Central Class: CIM_SPIPortt

Scoping Class: a CIM_System in a separate autonomous profile

Models a parallel SCSI port,

8.2 Description

This port represents a SCSI Parallel Interface (SPI).

8.3 Implementation

Because of addressing limits, the port may use multiple SCSI IDs to extend the addressing. The LUN Mapping/Masking common subprofile is not used with this port type. Figure 8 shows an SPI Target Port Instance.

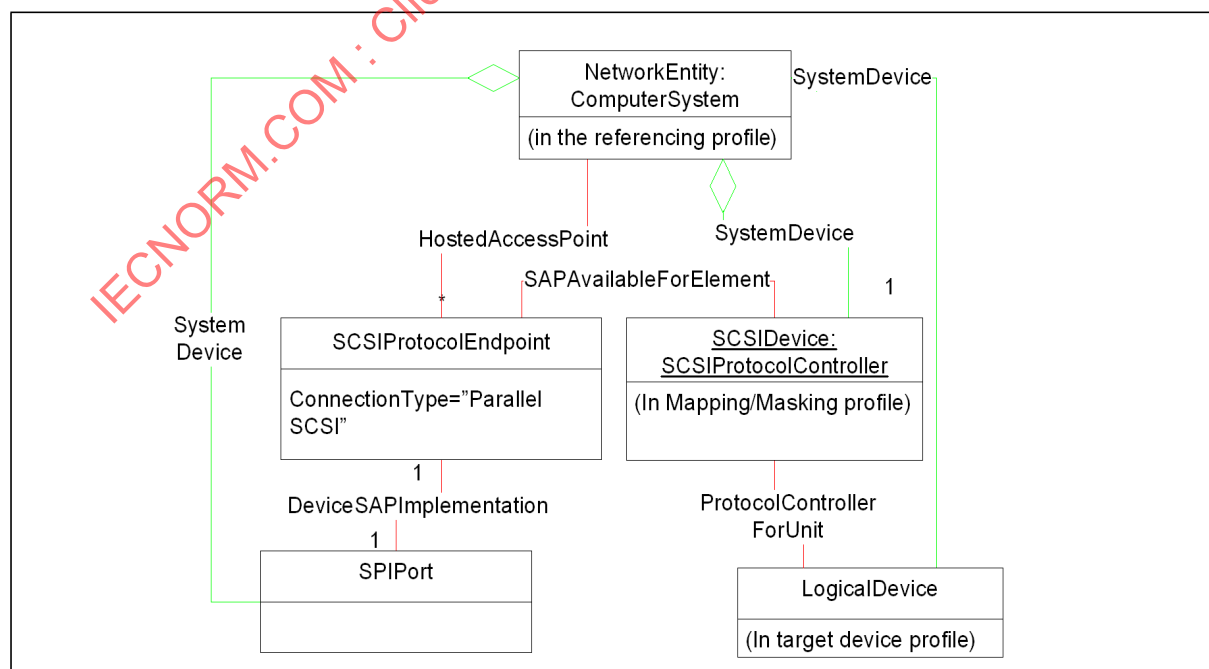


Figure 8 - SPI Target Port Instance Diagram

The SCSIProtocolEndpoint.ConnectionType shall be set to "Parallel SCSI". The SCSIProtocolEndpoint class is connected to a SPIPort. Attributes of SPIPort define the bus width and speed. The port class inherits the UsageRestriction attribute from LogicalPort. This attribute shall be set to "Front-end only"

8.4 Health and Fault Management

Table 10 shows SPIPort OperationalStatus.

Table 10 - SPIPort OperationalStatus

OperationalStatus	Description
OK	Port is online
Error	Port has a failure
Stopped	Port is disabled
InService	Port is in Self Test
Unknown	

8.5 Methods – Extrinsic Methods of this Subprofile

None

8.6 CIM elements

8.6.1 General

Table 11 describes the CIM elements for SPI Target Ports.

Table 11 - CIM Elements for SPI Target Ports

Element Name	Requirement	Description
8.6.2 CIM_DeviceSAPImplementation	Mandatory	Associates front-end SPIPort and target SCSIProtocolEndpoint.
8.6.3 CIM_HostedAccessPoint	Mandatory	Associates ComputerSystem to SCSIProtocolEndpoint.
8.6.4 CIM_SCSIProtocolEndpoint	Mandatory	Represents management characteristics related to the SCSI command set.
8.6.5 CIM_SPIPort	Mandatory	Represents the logical aspects of the physical port and may have multiple associated protocols.
8.6.6 CIM_SystemDevice (Port)	Mandatory	Associates ComputerSystem to LogicalPort.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_SPIPort	Mandatory	CQL -Create SPIPort.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_SPIPort AND SourceInstance.CIM_SPIPort::OperationalStatus <> PreviousInstance.SAS_Port::OperationalStatus	Mandatory	CQL -Modify SPIPort.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_SPIPort	Mandatory	CQL -Delete SPIPort.

8.6.2 CIM_DeviceSAPImplementation

Associates front-end SPIPort and target SCSIProtocolEndpoint. Limit to target ProtocolEndpoints and front-end ports. The class definition specializes the CIM_DeviceSAPImplementation definition in the Generic Target Ports profile. Properties or

methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 12 describes class CIM_DeviceSAPImplementation.

Table 12 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Requirement	Description and Notes
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint.
Antecedent (overridden)	Mandatory	Reference to SPIPort.

8.6.3 CIM_HostedAccessPoint

Associates ComputerSystem to SCSIProtocolEndpoint. Limit to targets (Role = 3). The class definition specializes the CIM_HostedAccessPoint definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 13 describes class CIM_HostedAccessPoint.

Table 13 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in the referencing profile.
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint.

8.6.4 CIM_SCSIProtocolEndpoint

Represents management characteristics related to the SCSI command set. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 14 describes class CIM_SCSIProtocolEndpoint.

Table 14 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType	Mandatory	Shall be 1 (Other).
OtherTypeDescription (overridden)	Mandatory	Shall be the string 'SCSI'.
ConnectionType (added)	Mandatory	Shall be 3 (Parallel SCSI).
Role (added)	Mandatory	Shall be 3 (Target).

8.6.5 CIM_SPIPort

Represents the logical aspects of the physical port and may have multiple associated protocols. The class definition specializes the CIM_LogicalPort definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 15 describes class CIM_SPIPort.

Table 15 - SMI Referenced Properties/Methods for CIM_SPIPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus	Mandatory	
UsageRestriction (overridden)	Mandatory	Shall be 2 (Front-end Only).
PortType (overridden)	Mandatory	Shall be 140 (SCSI Parallel).

8.6.6 CIM_SystemDevice (Port)

Associates ComputerSystem to LogicalPort.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 16 describes class CIM_SystemDevice (Port).

Table 16 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to ComputerSystem in the referencing profile.
PartComponent	Mandatory	Reference to Port.

EXPERIMENTAL

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

STABLE

9 FC Target Ports Profile

9.1 Synopsis

Profile Name: FC Target Ports (Component Profile)

Version: 1.4.0

Organization: SNIA

CIM Schema Version: 2.9.0

Table 17 describes the related profiles for FC Target Ports.

Table 17 - Related Profiles for FC Target Ports

Profile Name	Organization	Version	Requirement	Description
Indication	SNIA	1.5.0	Mandatory	

Central Class: CIM_FCPort

Scoping Class: a CIM_ComputerSystem in a referencing autonomous profile

9.2 Description

The FC Target Port Subprofile models the Fibre Channel specific aspects of a target storage system.

9.3 Implementation

9.3.1 General

For Fibre Channel ports, the concrete subclass of LogicalPort is FCPort. FCPort is always associated 1-1 with a SCSIProtocolEndpoint instance.

9.3.2 SMI-S 1.0 backwards compatibility

SCSIProtocolEndpoint was introduced in SMI-S 1.1.0 to enable support for non-FC transports and for non-SCSI protocols. In SMI-S 1.0, FCPort was associated directly to SCSIProtocolController. SCSIProtocolEndpoint, DeviceSAPImplementation, and SAPAvailableForElement are required and are used consistently across all target port subprofiles. To maintain backwards compatibility, ProtocolControllerForPort is still required in this version of SMI-S. But this association will be removed in a future versions and clients should start using the newer model. Figure 9 illustrates a Target Port instance.

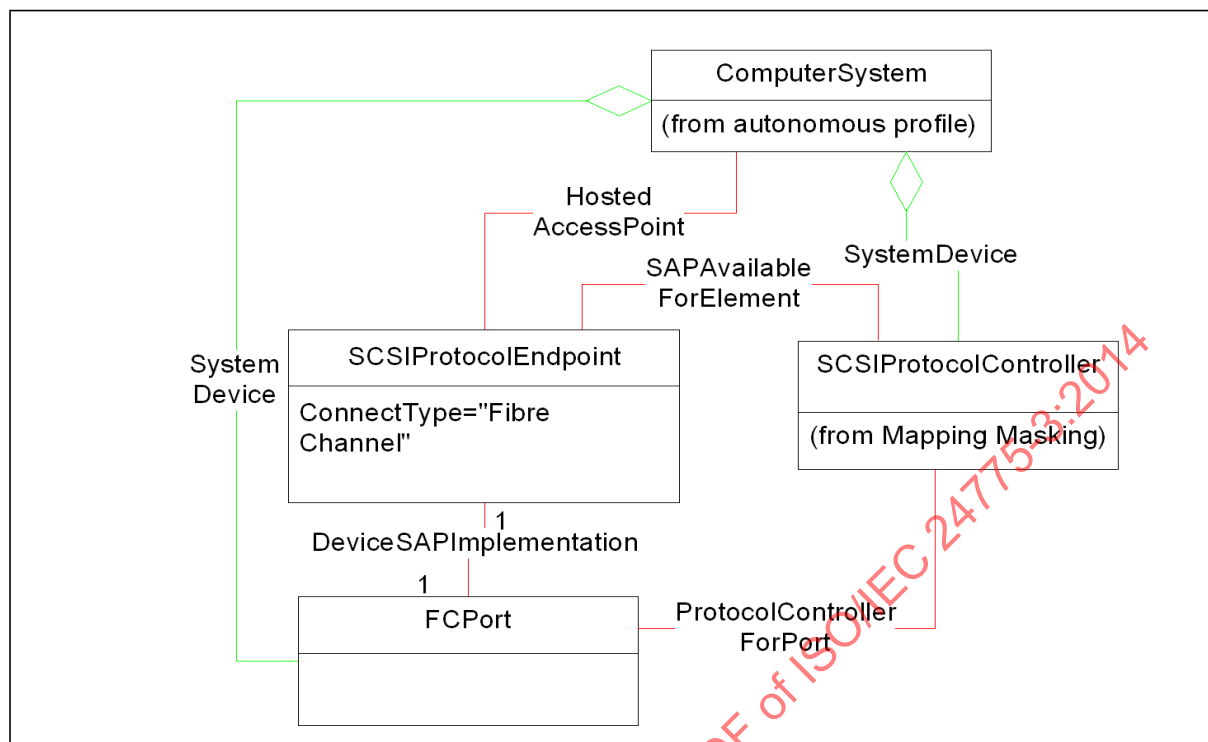


Figure 9 - FC Target Port Instance Diagram

9.4 Durable Names and Correlatable IDs of the Subprofile

FCPort.PermanantAddress shall contain the port's Port WWN.

9.5 Health and Fault Management

Table 18 describes FCPort OperationalStatus.

Table 18 - FCPort OperationalStatus

OperationalStatus	Description
OK	Port is online
Error	Port has a failure
Stopped	Port is disabled
InService	Port is in Self Test
Unknown	

9.6 Supported Profiles and Packages

None

9.7 Extrinsic Methods of this Subprofile

None

9.8 Client Considerations and Recipes

None

9.9 CIM Elements

9.9.1 General

Table 19 describes the CIM elements for FC Target Ports.

Table 19 - CIM Elements for FC Target Ports

Element Name	Requirement	Description
9.9.2 CIM_DeviceSAPImplementation	Mandatory	Associates FCPort and SCSIProtocolEndpoint.
9.9.3 CIM_FCPort	Mandatory	Represents the logical aspects of the physical port and may have multiple associated protocols.
9.9.4 CIM_HostedAccessPoint	Mandatory	Associates ComputerSystem to SCSIProtocolEndpoint.
9.9.5 CIM_ProtocolControllerForPort	Conditional	Conditional requirement: Support for the Masking and Mapping profile. Only required if the instrumentation claims compatibility with 1.0.
9.9.6 CIM_SCSIProtocolEndpoint	Mandatory	Represents management characteristics related to the SCSI command set.
9.9.7 CIM_SystemDevice (Port)	Mandatory	Associates controller ComputerSystem to FCPort.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_FCPort	Mandatory	Create FCPort.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.OperationalStatus <> PreviousInstance.OperationalStatus	Mandatory	Deprecated WQL -Change to FCPort OperationalStatus.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.CIM_FCPort::OperationalStatus <> PreviousInstance.CIM_FCPort::OperationalStatus	Mandatory	CQL -Change to FCPort OperationalStatus.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.Speed <> PreviousInstance.Speed	Optional	Deprecated WQL -Change to FCPort Speed.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.CIM_FCPort::Speed <> PreviousInstance.CIM_FCPort::Speed	Optional	CQL -Change to FCPort Speed.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.NetworkAddresses <> PreviousInstance.NetworkAddresses	Optional	Deprecated WQL -Change to FCPort NetworkAddresses.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.CIM_FCPort::NetworkAddresses <> PreviousInstance.CIM_FCPort::NetworkAddresses	Optional	CQL -Change to FCPort NetworkAddresses.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_FCPort	Mandatory	Delete FCPort.

9.9.2 CIM_DeviceSAPImplementation

Associates FCPort and SCSIProtocolEndpoint. The class definition specializes the CIM_DeviceSAPImplementation definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 20 describes class CIM_DeviceSAPImplementation.

Table 20 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Requirement	Description and Notes
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint.
Antecedent (overridden)	Mandatory	Reference to FCPort.

9.9.3 CIM_FCPort

Represents the logical aspects of the physical port and may have multiple associated protocols. The class definition specializes the CIM_LogicalPort definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 21 describes class CIM_FCPort.

Table 21 - SMI Referenced Properties/Methods for CIM_FCPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus	Mandatory	
UsageRestriction	Mandatory	Shall be 2 for ports restricted to Front-end only or 4 if the port is unrestricted.
PortType (overridden)	Mandatory	Shall be 0 1 10 11 12 13 14 15 16 17 18 (Unknown or Other or N or NL or F/NL or Nx or E or F or FL or B or G).
PermanentAddress (added)	Mandatory	Port WWN. Shall be 16 unseparated uppercase hex digits.
SupportedCOS (added)	Optional	
ActiveCOS (added)	Optional	
SupportedFC4Types (added)	Optional	
ActiveFC4Types (added)	Optional	

Table 21 - SMI Referenced Properties/Methods for CIM_FCPort

Properties	Requirement	Description and Notes
Speed (added)	Optional	Speed in bits per second. Shall be 0, 1062500000 (1GFC), 2125000000 (2GFC), 4250000000 (4GFC), 8500000000 (8GFC), 10518750000 (10GFC), 14025000000 (16GFC), 21037500000 (20GFC) or 28500000000 (32GFC).
MaxSpeed (added)	Optional	Maximum Port Speed.
NetworkAddresses (added)	Optional	For Fibre Channel end device ports, the Fibre Channel ID. Shall be 16 un-separated upper case hex digits.

9.9.4 CIM_HostedAccessPoint

Associates ComputerSystem to SCSIProtocolEndpoint. Limit to targets. The class definition specializes the CIM_HostedAccessPoint definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Create By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 22 describes class CIM_HostedAccessPoint.

Table 22 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint

Properties	Requirement	Description and Notes
Antecedent (overridden)	Mandatory	
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint.

9.9.5 CIM_ProtocolControllerForPort

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for the Masking and Mapping profile.

Table 23 describes class CIM_ProtocolControllerForPort.

Table 23 - SMI Referenced Properties/Methods for CIM_ProtocolControllerForPort

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to SCSIProtocolController.
Dependent	Mandatory	Reference to FCPort.

9.9.6 CIM_SCSIProtocolEndpoint

Represents management characteristics related to the SCSI command set. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 24 describes class CIM_SCSIProtocolEndpoint.

Table 24 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType	Mandatory	Shall be 1 (Other).
OtherTypeDescription (overridden)	Mandatory	Shall be the string 'SCSI'.
ConnectionType (added)	Mandatory	Shall be 2 (Fibre Channel).
Role (added)	Mandatory	Shall be 3 (Target) or 4 (Both Initiator and Target).

9.9.7 CIM_SystemDevice (Port)

Associates controller ComputerSystem to FCPort. The class definition specializes the CIM_SystemDevice definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 25 describes class CIM_SystemDevice (Port).

Table 25 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port)

Properties	Requirement	Description and Notes
GroupComponent (overridden)	Mandatory	Reference to ComputerSystem in the referencing profile.
PartComponent (overridden)	Mandatory	Reference to FCPort.

STABLE

STABLE

10 iSCSI Target Ports Subprofile

10.1 Synopsis

Profile Name: iSCSI Target Ports (Component Profile)

Version: 1.2.0

Organization: SNIA

CIM Schema Version: 2.13.1

Related Profiles for iSCSI Target Ports: Not defined in this International Standard.

Central Class: CIM_EthernetPortt

Scoping Class: a CIM_System in a separate autonomous profile

Models an iSCSI target port,

10.2 Description

The iSCSI Target Ports Subprofile describes the iSCSI specific aspects of a target device.

10.3 Implementation

10.3.1 General

iSCSI terminology is different than that used in other parts of SMI-S. Table 26 provides a map of terminology from iSCSI standards and CIM class names used in this International Standard. iSCSI does have a specific naming requirement for SCSIProtocolController that is described in Table 26.

Table 26 - iSCSI Terminology and SMI-S Class Names

iSCSI Term	CIM Class Name	Notes
Network Entity	ComputerSystem	The Network Entity represents a device or gateway that is accessible from the IP network. A Network Entity shall have one or more Network Portals, each of which can be used to gain access to the IP network by some iSCSI Nodes contained in that Network Entity.
Session	iSCSI Session	The group of TCP connections that link a Target with a target form a session (loosely equivalent to a SCSI I-T nexus). TCP connections can be added and removed from a session. Across all connections within a session, a Target sees one and the same target.
Connection	NetworkPipe	A connection is a TCP connection. Communication between the Target and target occurs over one or more TCP connections. The TCP connections carry control messages, SCSI commands, parameters, and data within iSCSI Protocol Data Units (iSCSI PDUs).
SCSI Port	iSCSIProtocolEndpoint	A SCSI Port using an iSCSI service delivery subsystem. A collection of Network Portals that together act as a SCSI Target or target.

Table 26 - iSCSI Terminology and SMI-S Class Names

Portal Group	SystemSpecificCollection	iSCSI supports multiple connections within the same session; some implementations will have the ability to combine connections in a session across multiple Network Portals. A Portal Group defines a set of Network Portals within an iSCSI Network Entity that collectively supports the capability of coordinating a session with connections spanning these portals. Not all Network Portals within a Portal Group need participate in every session connected through that Portal Group. One or more Portal Groups may provide access to an iSCSI Node. Each Network Portal, as utilized by a given iSCSI Node, belongs to exactly one portal group within that node.
Network Portal	TCPProtocolEndpoint, IPProtocolEndpoint, EthernetPort	The Network Portal is a component of a Network Entity that has a TCP/IP network address and that may be used by an iSCSI Node within that Network Entity for the connection(s) within one of its iSCSI sessions. A Network Portal in a Target is identified by its IP address. A Network Portal in a target is identified by its IP address and its listening TCP port.
Node	SCSIProtocolController	The iSCSI Node represents a single iSCSI Target or iSCSI target. There are one or more iSCSI Nodes within a Network Entity. The iSCSI Node is accessible via one or more Network Portals. An iSCSI Node is identified by its iSCSI Name. The separation of the iSCSI Name from the addresses used by and for the iSCSI Node allows multiple iSCSI nodes to use the same address, and the same iSCSI node to use multiple addresses.

Figure 10 is a class diagram for iSCSI Target Ports and uses the UML instance naming notation (InstanceName:ClassName) with the iSCSI-style names before the CIM names. Figure 26 explains the use of all these objects.

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

10.3.3 Settings

An iSCSI Session is established between a Target Port and a Target Port through the establishment of an initial iSCSI Connection, which happens during the “Leading” Login. At this time the operational properties for the Session are negotiated and also the operational properties for the initial Connection. Additional Connections for the Session are established through subsequent logins. For many operational properties both the Target and Target have settings that specify the starting position for the negotiation process. The settings for negotiating Session-wide operational properties (found in iSCSISession) are in iSCSISessionSettings. Likewise the settings for negotiating Connection level operational properties (found in iSCSI Connection) are in iSCSIConnectionSettings. For example, iSCSISessionSettings contains the property MaxConnectionsPerSession, which is the value that the local system (which in this subprofile is the Target) would like to use for Session. When the leading login is complete the actual value agreed upon with the Target is in the property MaxConnectionsPerSession in iSCSI Session.

Different implementations may scope the settings classes differently.

iSCSISessionSettings can be associated to any one of the following classes:

- iSCSIProtocolEndpoint: The Settings apply to Sessions created on the iSCSI Port represented by the iSCSIProtocolEndpoint.
- SCSIProtocolController: The Settings apply to Sessions created on all iSCSIProtocolEndpoint belonging to the iSCSI Node represented by the SCSIProtocolController.
- ComputerSystem: The Settings apply to Sessions created on all iSCSIProtocolEndpoints belonging to all SCSIProtocolControllers belonging to the ComputerSystem.

iSCSIConnectionSettings can be associated to any one of the following classes:

- TCPProtocolEndpoint: The Settings apply to each Connection created using the Network Portal represented by the TCPProtocolEndpoint, regardless of which iSCSIProtocolEndpoint owns the Session that the Connection belongs to.
- iSCSIProtocolEndpoint: The Settings apply to Connections using NetworkPortals to which the iSCSIProtocolEndpoint is bound and belonging to Sessions on that same iSCSIProtocolEndpoint.

10.3.4 Durable Names and Correlatable IDs of the Subprofile

The Name property for the iSCSI node (SCSIProtocolController) shall be a compliant iSCSI name as described in *Information Technology – Storage Management, Part 2 Common Architecture* 8.8. NameFormat shall be set to “iSCSI Name”.

The Name property for iSCSIProtocolEndpoint shall be a compliant iSCSI name as described in *Information Technology – Storage Management, Part 2 Common Architecture* 8.8. ConnectionType shall be set to “iSCSI”.

10.4 Health and Fault Management

Table 27 defines the SMI-S-defined meanings of the OperationalStatus property for EthernetPort used in the SB Target Port Profile.

Table 27 - EthernetPort OperationalStatus

OperationalStatus	Description
OK	Port is online
Error	Port has a failure
Stopped	Port is disabled
InService	Port is in Self Test
Unknown	

10.5 Supported Subprofiles and Packages

None

10.6 Methods of this Subprofile

10.6.1 General

The iSCSIConfigurationService provides the following methods that allow a client to manipulate iSCSIProtocolEndpoints in an iSCSI Target Node. The class iSCSIProtocolController models the iSCSI Target Port. The instance of the service is scoped by an instance of ComputerSystem that represents that Network Entity. The capabilities of this service are defined in the companion class iSCSIConfigurationCapabilities.

10.6.2 CreateiSCSINode

10.6.2.1 General

This method creates an iSCSI Node in the form of an instance of SCSIProtocolController. As part of the creation process a SystemDevice association is created between the new SCSIProtocolController and the scoping Network Entity (ComputerSystem) hosting this service.

CreateiSCSINode

IN, string **Alias,**

The iSCSI Alias for the new Node.

OUT, SCSIProtocolController REF **iSCSINode,**

A reference to the new SCSIProtocolController that is created.

10.6.2.2 Return Values

Success

Not Supported

Unspecified Error

Timeout

Failed

Node Creation Not Supported

Alias in use by Other Node

10.6.2.3 Created Instances

SCSIProtocolController

SystemDevice

10.6.2.4 Deleted Instances

None

10.6.2.5 Modified Instances

None

10.6.3 DeleteiSCSINode**10.6.3.1 General**

The method deletes an instance of SCSIProtocolController representing an iSCSI Node and all associations in which this SCSIProtocolController is referenced. If Sessions are active on iSCSIProtocolEndpoints belonging to this Node an error will be returned. If no Sessions are active the scoped iSCSIProtocolEndpoints will be deleted.

DeleteiSCSINode

IN, SCSIProtocolController REF **iSCSINode**

The SCSIProtocolController to be deleted.

10.6.3.2 Return Values

Success

Not Supported

Unspecified Error

Timeout

Failed

Invalid Parameter

SCSIProtocolController Non-existent

Sessions Active on Node Ports

10.6.3.3 Created Instances

None

10.6.3.4 Deleted Instances

SCSIProtocolController

SystemDevice

iSCSIProtocolEndpoint

HostedAccessPoint

SAPAvailableForElement

BindsTo

10.6.3.5 Modified Instances

None

10.6.4 CreateiSCSIProtocolEndpoint**10.6.4.1 General**

This method creates an iSCSI Port in the form of an instance of iSCSIProtocolEndpoint. As part of the creation process the iSCSIProtocolEndpoint is 'bound to' the underlying TCPProtocolEndpoints which are specified as inputs by creating instances of the BindsTo association between the new instance and those instances. In addition, an instance of SAPAvailableForElement is created between the specified SCSIProtocolController and the new instance of iSCSIProtocolEndpoint.

CreateiSCSIProtocolEndpoint

IN, SCSIProtocolController REF **iSCSINode**,

The SCSIProtocolController instance representing the iSCSI Node that will contain the iSCSI Port.

IN, uint16 **Role**,

For iSCSI, each iSCSIProtocolEndpoint acts as either a target or a Target endpoint. This property indicates which role this iSCSIProtocolEndpoint implements.

IN, string **Identifier**,

The Identifier shall contain the Target Portal Group Tag (TGPT). Each iSCSIProtocolEndpoint (iSCSI port) associated to a common SCSIProtocolController (iSCSI node) has a unique Identifier. This field is a string that contains 12 hexadecimal digits. If the property IdentifierSelectionSupported in class iSCSIConfigurationCapabilities is false, this parameter shall be set to NULL.

IN, ProtocolEndpoint REF **NetworkPortals[]**,

An Array of References to TCPProtocolEndpoints representing Target Network Portals. The TCPProtocolEndpoints specified each shall be associated to an instance of IPProtocolEndpoint via a BindsTo association in order to provide the Target Network Portal functionality. The selected Portal endpoints shall be from the same SystemSpecificCollection, which represents a Portal Group.

OUT, iSCSIProtocolEndpoint REF **iSCSIPort**,

A reference to the new iSCSIProtocolEndpoint that is created.

10.6.4.2 Return Values

Success

Not Supported

Unspecified Error

Timeout

Failed

SCSIProtocolController Non-existent

Role Not Supported By Specified SCSIProtocolController

Identifier In Use, Not Unique

Identifier Selection Not Supported

ProtocolEndpoint Non-Existent

TCPProtocolEndpoint Not Bound To Underlying IPProtocolEndpoint

TCPProtocolEndpoint In Use By Other iSCSIProtocolEndpoint In Same Target SCSIProtocolController.

ProtocolEndpoints Not From Same Endpoint Collection

10.6.4.3 Created Instances

iSCSIProtocolEndpoint

HostedAccessPoint

SAPAvailableForElement

Bindsto

10.6.4.4 Deleted Instances

None

10.6.4.5 Modified Instances

None

10.6.5 DeleteiSCSIProtocolEndpoint

10.6.5.1 General

The method deletes an instance of iSCSIProtocolEndpoint and all associations in which this iSCSIProtocolEndpoint is referenced.

DeleteiSCSIProtocolEndpoint

IN, iSCSIProtocolEndpoint REF iSCSIPort

The iSCSIProtocolEndpoint to be deleted.

10.6.5.2 Return Values

Success

Not Supported

Unspecified Error

Timeout

Failed

Invalid Parameter

Endpoint Non-existent

10.6.5.3 Created Instances

None

10.6.5.4 Deleted Instances

iSCSIProtocolEndpoint

HostedAccessPoint

SAPAvailableForElement

Bindsto

10.6.5.5 Modified Instances

None

10.6.6 BindiSCSIProtocolEndpoint

10.6.6.1 General

This method provides for modification of an existing iSCSI Port by associating a TCPProtocolEndpoint representing a Target Network Portal to the iSCSIProtocolEndpoint. The association is persisted as an instance of Bindsto. The selected Portal endpoint shall be from the same SystemSpecificCollection, which represents a Portal Group, as those endpoints currently bound to the iSCSIProtocolEndpoint.

This action is intended to be reversed by the use of the intrinsic method '**DeleteInstance**'.

BindiSCSIProtocolEndPoint

IN, iSCSIProtocolEndpoint REF **iSCSIPort**,

A reference to the iSCSIProtocolEndpoint

IN, ProtocolEndpoint REF **NetworkPortal**

An instance of TCPProtocolEndpoint representing the Network Portal to be added

10.6.6.2 Return Values

Success

Not Supported

Unspecified Error

Timeout

Failed

Invalid Parameter

ProtocolEndpoint Non-Existent

TCPProtocolEndpoint Not Bound To Underlying IPProtocolEndpoint

ProtocolEndpoint In Use By Other iSCSIProtocolEndpoint In Same Target
SCSIProtocolController

ProtocolEndpoint Not From Same Endpoint Collection

10.6.6.3 Created Instances

BindsTo

10.6.6.4 Deleted Instances

None

10.6.6.5 Modified Instances

None

10.7 Client Considerations and Recipes

10.7.1 Discover the iSCSI Target Port capabilities

```
// DESCRIPTION
// Discover the iSCSI Target Port capabilities.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. The ComputerSystem representing the target system of interest has
//    been
//    previously identified and defined in the $NetworkEntity-> variable.

// MAIN
// Step 1. Locate the instance of CIM_iSCSICapabilities associated to the
// target ComputerSystem.
$iSCSICapabilities[] = Associators($NetworkEntity->,
    "CIM_ElementCapabilities",
    "CIM_iSCSICapabilities",
    "ManagedElement",
```

```

        "Capabilities",
        {"MinimumSpecificationVersionSupported",
        "MaximumSpecificationVersionSupported",
        "AuthenticationMethodsSupported"})

if ($iSCSICapabilities[] == null || $iSCSICapabilities[].length != 1) {
    <ERROR! The iSCSI capabilities could not be found>
}
$Capabilities = $iSCSICapabilities[0]

```

10.7.2 Identify the iSCSI Nodes in a target system

```

// DESCRIPTION
//
// Identify the iSCSI Nodes in a target system.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. The ComputerSystem representing the Network Entity of interest has
//     been
// previously identified and defined in the $NetworkEntity-> variable.

// MAIN
// Step 1. Locate the instances of CIM_SCSIProtocolController with a
//     NameFormat
// property value of "iSCSI Name".
$ProtocolControllers[] = Associators($NetworkEntity->,
    "CIM_SystemDevice",
    "CIM_SCSIProtocolController",
    "GroupComponent",
    "PartComponent",
    false,
    false,
    {"Name", "NameFormat"})

// Step 2. Locate the SCSIProtocolControllers that represent the iSCSI
//     Nodes.
$iSCSINodes[]
#index = 0
for (#i in $ProtocolControllers[]) {
    if ($ProtocolControllers[#i].NameFormat == "iSCSI Name") {
        // Filter out SCSIProtocolControllers previously encountered.
        if (!contains($ProtocolControllers[#i].Name, #NodeNames[])) {
            #NodeNames[#index] = $ProtocolControllers[#i].Name
            $iSCSINodes[#index++] = $ProtocolControllers[#i]
        }
    }
}
}
<EXIT: $Nodes[] contains the results>

```

10.7.3 Identify the iSCSI Ports on an given iSCSI node

```

// DESCRIPTION

```



```
// Identify the iSCSI Ports on an given iSCSI node.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. The SCSIProtocolController representing an iSCSI Node of interest
//    has
// been previously identified and defined in the $iSCSINode-> variable.

// This function returns the instance(s) of iSCSI ports on the specified
// iSCSI node, or null if none are found.
sub $iSCSIPorts[] getiSCSIPortsOnNode($Node->) {

    // Step 1. Locate the iSCSI Ports, which are represented by instances
    // of
    // iSCSIProtocolEndpoint, on the iSCSI Node of interest.
    $iSCSIPorts[] = Associators($iSCSINode->,
        "CIM_SAPAvailableForElement",
        "CIM_iSCSIProtocolEndpoint",
        "ManagedElement",
        "AvailableSAP",
        false,
        false,
        {"Name", "Identifier", "Role"})

    if ($iSCSIPorts[].length == 0) {
        return (null)
    }
    return ($iSCSIPorts[])
}

// MAIN
$iSCSIPorts[] = &getiSCSIPortsOnNode($iSCSINode->)
```

10.7.4 Identify the iSCSI sessions existing on an iSCSI node

```
// DESCRIPTION
// Identify the iSCSI sessions existing on an iSCSI node.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. The SCSIProtocolController representing the iSCSI Node of interest
//    has
// been previously identified and defined in the $iSCSINode-> variable

// Step 1. Retrieve the CIM_iSCSIProtocolEndpoints for an
// CIM_SCSIProtocolController representing a node.
$iSCSIPorts[] = @getiSCSIPortsOnNode($iSCSINode->)
if ($iSCSIPorts[] == null) {
    <ERROR! No iSCSI ports located on the specified iSCSI node>
}

// Step 2. Retrieve the iSCSI session associated with each iSCSI port.
$iSCSISessions[]
#index = 0
```

```

#PropList[] = {"Directionality", "SessionType", "TSIH", "EndPointName",
    "CurrentConnections", "InitialR2T", "ImmediateData",
    "MaxOutstandingR2T", "MaxUnsolicitedFirstDataBurstLength",
    "MaxDataBurstLength", "AuthenticationMethodUsed",
    "DataSequenceInOrder", "DataPDUInOrder", "ErrorRecoveryLevel"}
for (#i in $iSCSIPorts[]) {
    $Sessions[] = Associators($iSCSIPorts[#i].getObjectPath(),
        "CIM_EndpointOfNetworkPipe",
        "CIM_iSCSISession",
        "Antecedent",
        "Dependent",
        #PropList[])
    if ($Sessions[] != null && $Sessions[].length == 1) {
        $iSCSISessions[#index++] = $Sessions[0]
    }
}
<EXIT: $iSCSISessions[] contains the iSCSI Sessions>

```

10.7.5 Create an iSCSI Target Node on an iSCSI Network Entity

```

// DESCRIPTION
// Create an iSCSI Target Node on an iSCSI Network Entity
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. The ComputerSystem representing the Network Entity of interest has
//    been
//    previously identified and defined in the $NetworkEntity-> variable.

// MAIN
// Step 1. Locate the CIM_iSCSIConfigurationService hosted by the System.
$iSCSIConfigurationService->[] = AssociatorNames($NetworkEntity->,
    "CIM_HostedService",
    "CIM_iSCSIConfigurationService",
    "Antecedent",
    "Dependent")
if ($iSCSIConfigurationService->[] == null ||
    $iSCSIConfigurationService->[].length == 0) {
    <ERROR! Required iSCSI Configuration Service not available>
}

// Step 2. Examine the capabilities to determine if Node creation is
//    supported.
$ConfigurationCapabilities[] = Associators($iSCSIConfigurationService->
    >[0],
    "CIM_ElementCapabilities",
    "CIM_iSCSIConfigurationCapabilities",
    "ManagedElement",
    "Capabilities",
    false,
    false,
    {"iSCSINodeCreationSupported "})
if ($ConfigurationCapabilities[] == null ||

```

```
$ConfigurationCapabilities[].length == 0) {  
  <ERROR! Required iSCSI Configuration Service capabilities not  
    available>  
}  
  
// Step 3. Create the iSCSI Target Node if supported by the device.  
if ($ConfigurationCapabilities[0].iSCSINodeCreationSupported == true) {  
  %InArguments["Alias"] = "Some Target Alias"  
  #ReturnValue = invokeMethod($iSCSIConfigurationService->[0],  
    "CreateiSCSINode",  
    %InArguments[],  
    %OutArguments[])  
  
  if (#ReturnValue == 0) {  
    $NewNode-> = $OutArguments["iSCSINode"]  
    <EXIT: The node was created>  
  } else {  
    <EXIT: The method returned an error; the Node was not created>  
  }  
} else {  
  <EXIT: Node Creation is not supported>  
}
```

10.7.6 Create an iSCSI Target Port on an iSCSI target node

```
// DESCRIPTION  
// This recipe describes how to create an iSCSI Target Port on an iSCSI  
// target  
// node.  
//  
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS  
// 1. The object name for the ComputerSystem representing the Network  
// Entity of  
// interest has been previously identified and defined in the  
// $NetworkEntity->  
// variable.  
// 2. The object name for the SCSIProtocolController representing the  
// iSCSI Node  
// within which to create the iSCSI Port has been identified and defined  
// in the // $Node-> variable.  
// 3. The object names for one or more TCPProtocolEndpoints representing  
// Target  
// Network Portals have been previously identified and defined in the  
// Portals->[] array variable.  
  
// MAIN  
// Step 1. Find a CIM_iSCSIConfigurationService associated to  
// ComputerSystem  
// by HostedService.  
$iSCSIConfigurationService->[] = AssociatorNames($NetworkEntity->,  
  "CIM_HostedService",  
  "CIM_iSCSIConfigurationService",  
  "Antecedent",
```

```

    "Dependent")

// Step 2. Examine the associated CIM_iSCSIConfigurationCapabilities to
// determine if Target Port manipulation is supported.
$ConfigurationCapabilities[] = Associators($iSCSIConfigurationService-
    >[0],
    "CIM_ElementCapabilities",
    "CIM_iSCSIConfigurationCapabilities",
    "ManagedElement",
    "Capabilities",
    false,
    false,
    {"iSCSIProtocolEndpointCreationSupported"})

// Step 3. Given an instance of CIM_SCSIProtocolController representing a
// Node($Node->), and one or more TCPProtocolEndpoints representing
// Target
// Network Portals(Portals->[]), invoke the method
// CreateiSCSIProtocolEndpoint
// to create the iSCSIProtocolEndpoint.
if ($ConfigurationCapabilities[0].iSCSIProtocolEndpointCreationSupported
    == true) {

    %InArguments["iSCSINode"] = $Node->
    %InArguments["Role"] = 3// "Target"
    %InArguments["NetworkPortals"] = Portals->[]

    #ReturnValue = InvokeMethod($iSCSIConfigurationService->[0],
        "CreateiSCSIProtocolEndpoint",
        %InArguments[],
        %OutArguments[])

    if (#ReturnValue == 0) {
        $NewiSCSIProtocolEndpoint-> = $OutArguments["iSCSIPort"]
        <EXIT: The ProtocolEndpoint was created>
    } else {
        <EXIT: The method returned an error; the ProtocolEndpoint was not
            created>
    }
} else {
    <EXIT: iSCSIProtocolEndpoint creation is not supported>
}

```

10.7.7 Add a Network Portal to a Target Port

```

// DESCRIPTION
// This recipe describes how to add a Network Portal to a Target Port.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. The object name for the ComputerSystem representing the Network
// Entity of

```

```
// interest has been previously identified and defined in the
    $NetworkEntity->

// variable.
// 2. The object name for the instance of iSCSIProtocolEndpoint
    representing a
// Port has been previously identified and defined in the $iSCSIPort->
    variable.
// 3. The object name for the instance of TCPProtocolEndpoint
    representing a
// Target Network Portal has been previously identified and defined in
    the
// $Portal-> variable.

// MAIN
// Step 1. Find a CIM_iSCSIConfigurationService associated to
    ComputerSystem by // HostedService.
$iSCSIConfigurationService->[] = AssociatorNames($NetworkEntity->,
    "CIM_HostedService",
    "CIM_iSCSIConfigurationService",
    "Antecedent",
    "Dependent")

// Step 2. Examine the associated CIM_iSCSIConfigurationCapabilities to
// determine if Target Port manipulation is supported.
$ConfigurationCapabilities[] = Associators($iSCSIConfigurationService->
    >[0],
    "CIM_ElementCapabilities",
    "CIM_iSCSIConfigurationCapabilities",
    "ManagedElement",
    "Capabilities",
    false,
    false,
    {"iSCSIProtocolEndpointCreationSupported"})

// Step 3. Given an instance of CIM_iSCSIProtocolEndpoint representing a
// Port (iSCSIPort->), and an instance of TCPProtocolEndpoint
    representing a
// Target Network Portal($Portal->), invoke Bind iSCSIProtocolEndpoint().
if ($ConfigurationCapabilities[0].iSCSIProtocolEndpointCreationSupported
    == true) {

    %InArguments["iSCSIPort"] = $iSCSIPort->
    %InArguments["NetworkPortal"] = $Portal->

    #ReturnValue = invokeMethod($iSCSIConfigurationService->[0],
        "Bind iSCSIProtocolEndpoint",
        %InArguments[],
        %OutArguments[])

    if (#ReturnValue == 0) {
        <EXIT: The ProtocolEndpoint was modified>
    } else {
```

```

        <EXIT: The method returned an error; the ProtocolEndpoint was not
            modified>
    }
} else {
    <EXIT: iSCSIProtocolEndpoint modification is not supported>
}

```

10.7.8 Determine the health of Nodes in a target system

```

//
// DESCRIPTION
// Recipe ISCSI_TRGT08:
// Determine the health of Nodes in a target system.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. The object name for the SCSIProtocolController representing
// the iSCSI Node of interest has been previously identified and
// defined in the $iSCSINode-> variable

//
// Step 1.
// Given an instance of CIM_SCSIProtocolController($iSCSINode->) ,
// get the instances of CIM_iSCSI_SessionFailures and
// CIM_iSCSI_LoginStatistics associated by ElementStatisticalData.
//
$SessionFailures[] = Associators(
    $iSCSINode->,
    "CIM_ElementStatisticalData",
    "CIM_iSCSI_SessionFailures",
    "ManagedElement",
    "Stats" );

$LoginStatistics[] = Associators(
    $iSCSINode->,
    "CIM_ElementStatisticalData",
    "CIM_iSCSI_LoginStatistics",
    "ManagedElement",
    "Stats" );

<EXIT: The statistics are in $SessionFailures[0] and $LoginStatistics[0]
>

```

10.7.9 Determine the health of a Session on a target system

```

//
// DESCRIPTION
// Recipe ISCSI_TRGT09:
// Determine the health of a Session on a target system.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. The object name for the iSCSI_Session of interest has been
// previously identified and defined in the $iSCSI_Session-> variable.

```

```
// Step 1.
// Given an instance of CIM_iSCSI Session,
// get the instance of CIM_iSCSI SessionStatistics
// associated by ElementStatisticalData.
//
$SessionStatistics[] = Associators(
    $iSCSI Session->,
    "CIM_ElementStatisticalData",
    "CIM_iSCSI SessionStatistics",
    "ManagedElement",
    "Stats" );
```

```
<EXIT: The statistics are in $SessionStatistics[0]>
```

10.7.10 Configure the default settings for Sessions created in a target computer system

```
//
// DESCRIPTION
// Recipe iSCSI_TRGT10:
// Configure the default settings for Sessions created in a target
// computer system.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. The object name for the SCSIProtocolController representing the
// iSCSI Node of interest has been previously identified and defined
// in the $iSCSI Node-> variable.
//
//
// Step 1.
// Find and modify an instance of CIM_iSCSI SessionSettings associated
// to a ComputerSystem, CIM_SCSIProtocolController, or
// CIM_iSCSIProtocolEndpoint.
//
$SessionSettings[] = Associators(
    $iSCSIProtocolEndpoint->,
    "CIM_ElementSettingData",
    "CIM_iSCSI SessionSettings",
    "ManagedElement",
    "SettingData" );

#MaxConnectionsPerSession = 4;

$SessionSettings[0].MaxConnectionsPerSession =
    #MaxConnectionsPerSession;

$ModifyInstance(
    $SessionSettings[0],
    false,
```

```
{ "MaxConnectionsPerSession" } );
```

```
<EXIT: Success>
```

10.7.11 Configure default settings for Connections on Network Portals used by an iSCSI-ProtocolEndpoint

```
//
// DESCRIPTION
// Recipe iSCSI_TRGT11:
// Configure the default settings for iSCSI Connections created on
// Network Portals used by an iSCSIProtocolEndpoint.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. The object name for the iSCSI Session of interest has been
// previously identified and defined in the $iSCSISession->
// variable

//
// Step 1.
// Find and modify an instance of CIM_iSCSIConnectionSettings
// associated to a iSCSIProtocolEndpoint($iSCSIProtocolEndpoint->).
//
$ConnectionSettings[] = Associators(
    $iSCSIProtocolEndpoint->,
    "CIM_ElementSettingData",
    "CIM_iSCSIConnectionSettings",
    "ManagedElement",
    "SettingData" );

#MaxRecvDataSegLength = 4096;

$ConnectionSettings[0].MaxReceiveDataSegmentLength =
    #MaxRecvDataSegLength;

$ModifyInstance(
    $ConnectionSettings[0],
    false,
    { "MaxReceiveDataSegmentLength" } );`

<EXIT: Success>
```

10.7.12 Get the statistics for a Session on a target system

The statistics are properties in the same class as the health information; see 10.7.9.

10.7.13 Configure Enable/disable header and data digest

See 10.7.11.

10.8 CIM Elements

10.8.1 General

Table 28 describes the CIM elements for iSCSI Target Ports.

Table 28 - CIM Elements for iSCSI Target Ports

Element Name	Requirement	Description
10.8.2 CIM_BindsTo (TCPProtocolEndpoint to IPProtocolEndpoint)	Mandatory	
10.8.3 CIM_BindsTo (iSCSIProtocolEndpoint to TCPProtocolEndpoint)	Mandatory	
10.8.4 CIM_ConcreteDependency	Mandatory	
10.8.5 CIM_DeviceSAPImplementation (EthernetPort to IPProtocolEndpoint)	Optional	
10.8.6 CIM_DeviceSAPImplementation (EthernetPort to iSCSIProtocolEndpoint)	Optional	
10.8.7 CIM_ElementCapabilities (iSCSIConfigurationCapabilities to System)	Mandatory	
10.8.8 CIM_ElementCapabilities (iSCSIConfigurationCapabilities to iSCSIConfigurationService)	Conditional	Conditional requirement: Active configuration is supported.
10.8.9 CIM_ElementSettingData (iSCSIConnectionSettings to TCPProtocolEndpoint)	Mandatory	
10.8.10 CIM_ElementSettingData (iSCSIConnectionSettings to iSCSIProtocolEndpoint)	Mandatory	
10.8.11 CIM_ElementSettingData (iSCSIConnectionSettings to SCSIProtocolController)	Mandatory	
10.8.12 CIM_ElementSettingData (iSCSIConnectionSettings to System)	Mandatory	
10.8.13 CIM_ElementSettingData (iSCSIConnectionSettings to iSCSIProtocolEndpoint)	Mandatory	
10.8.14 CIM_ElementStatisticalData (iSCSILoginStatistics to SCSIProtocolController)	Mandatory	
10.8.15 CIM_ElementStatisticalData (iSCSIConnectionFailures to SCSIProtocolController)	Mandatory	
10.8.16 CIM_ElementStatisticalData (iSCSIConnectionStatistics to iSCSIConnection)	Mandatory	
10.8.17 CIM_EndpointOfNetworkPipe (iSCSIConnection to TCPProtocolEndpoint)	Mandatory	
10.8.18 CIM_EndpointOfNetworkPipe (iSCSIConnection to iSCSIProtocolEndpoint)	Mandatory	
10.8.19 CIM_EthernetPort	Optional	
10.8.20 CIM_HostedAccessPoint (System to IPProtocolEndpoint)	Mandatory	
10.8.21 CIM_HostedAccessPoint (System to TCPProtocolEndpoint)	Mandatory	
10.8.22 CIM_HostedAccessPoint (System to iSCSIProtocolEndpoint)	Mandatory	
10.8.23 CIM_HostedCollection	Mandatory	
10.8.24 CIM_HostedService	Optional	

Table 28 - CIM Elements for iSCSI Target Ports

Element Name	Requirement	Description
10.8.25 CIM_IPProtocolEndpoint	Mandatory	
10.8.26 CIM_MemberOfCollection	Optional	
10.8.27 CIM_NetworkPipeComposition	Optional	
10.8.28 CIM_SAPAvailableForElement	Mandatory	
10.8.29 CIM_SCSIProtocolController	Mandatory	
10.8.30 CIM_SystemDevice (System to EthernetPort)	Mandatory	This association links all EthernetPorts to the scoping system.
10.8.31 CIM_SystemDevice (System to SCSIProtocolController)	Mandatory	This association links SCSIProtocolControllers to the scoping system.
10.8.32 CIM_SystemSpecificCollection	Optional	
10.8.33 CIM_TCPProtocolEndpoint	Mandatory	
10.8.34 CIM_iSCSICapabilities	Mandatory	
10.8.35 CIM_iSCSIConfigurationCapabilities	Conditional	Conditional requirement: Active configuration is supported.
10.8.36 CIM_iSCSIConfigurationService	Optional	
10.8.37 CIM_iSCSIConnection	Optional	
10.8.38 CIM_iSCSIConnectionSettings	Optional	
10.8.39 CIM_iSCSILoginStatistics	Optional	
10.8.40 CIM_iSCSIProtocolEndpoint	Mandatory	
10.8.41 CIM_iSCSISession	Mandatory	
10.8.42 CIM_iSCSISessionFailures	Optional	
10.8.43 CIM_iSCSISessionSettings	Mandatory	
10.8.44 CIM_iSCSISessionStatistics	Optional	
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_EthernetPort	Optional	Create EthernetPort.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_EthernetPort AND SourceInstance.CIM_EthernetPort::OperationalStatus <> PreviousInstance.CIM_EthernetPort::OperationalStatus	Optional	CQL -Modify EthernetPort.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_EthernetPort	Optional	Delete EthernetPort.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_iSCSIProtocolEndpoint	Mandatory	Create iSCSIProtocolEndpoint.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_iSCSIProtocolEndpoint	Mandatory	Delete SCSIProtocolEndpoint.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_SCSIProtocolController	Mandatory	Create SCSIProtocolController.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_SCSIProtocolController	Mandatory	Delete iSCSIProtocolController.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_iSCSISession	Optional	Create iSCSISession.

Table 28 - CIM Elements for iSCSI Target Ports

Element Name	Requirement	Description
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_iSCSISession AND SourceInstance.CIM_iSCSISession::CurrentConnections <> PreviousInstance.CIM_iSCSISession::CurrentConnections	Optional	CQL -Modify iSCSISession.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_iSCSISession AND SourceInstance.CurrentConnections <> PreviousInstance.CurrentConnections	Optional	Deprecated WQL -Modify iSCSISession.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_iSCSISession	Optional	Delete iSCSISession.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_iSCSIConnection	Optional	Create iSCSIConnection.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_iSCSIConnection	Optional	Delete iSCSIConnection.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_iSCSISessionSettings	Mandatory	Modify iSCSISessionSettings.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_iSCSIConnectionSettings	Optional	Modify iSCSIConnectionSettings.

10.8.2 CIM_BindsTo (TCPProtocolEndpoint to IPProtocolEndpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 29 describes class CIM_BindsTo (TCPProtocolEndpoint to IPProtocolEndpoint).

Table 29 - SMI Referenced Properties/Methods for CIM_BindsTo (TCPProtocolEndpoint to IPProtocolEndpoint)

Properties	Requirement	Description and Notes
Dependent	Mandatory	
Antecedent	Mandatory	

10.8.3 CIM_BindsTo (iSCSIProtocolEndpoint to TCPProtocolEndpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 30 describes class CIM_BindsTo (iSCSIProtocolEndpoint to TCPProtocolEndpoint).

Table 30 - SMI Referenced Properties/Methods for CIM_BindsTo (iSCSIProtocolEndpoint to TCPProtocolEndpoint)

Properties	Requirement	Description and Notes
Dependent	Mandatory	
Antecedent	Mandatory	

10.8.4 CIM_ConcreteDependency

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 31 describes class CIM_ConcreteDependency.

Table 31 - SMI Referenced Properties/Methods for CIM_ConcreteDependency

Properties	Requirement	Description and Notes
Dependent	Mandatory	
Antecedent	Mandatory	

10.8.5 CIM_DeviceSAPImplementation (EthernetPort to IPProtocolEndpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 32 describes class CIM_DeviceSAPImplementation (EthernetPort to IPProtocolEndpoint).

Table 32 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation (Ethernet-Port to IPProtocolEndpoint)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

10.8.6 CIM_DeviceSAPImplementation (EthernetPort to iSCSIProtocolEndpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 33 describes class CIM_DeviceSAPImplementation (EthernetPort to iSCSIProtocolEndpoint).

Table 33 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation (Ethernet-Port to iSCSIProtocolEndpoint)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

10.8.7 CIM_ElementCapabilities (iSCSIConfigurationCapabilities to System)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 34 describes class CIM_ElementCapabilities (iSCSIConfigurationCapabilities to System).

Table 34 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (iSCSIConfigurationCapabilities to System)

Properties	Requirement	Description and Notes
ManagedElement	Mandatory	
Capabilities	Mandatory	

10.8.8 CIM_ElementCapabilities (iSCSIConfigurationCapabilities to iSCSIConfiguration-Service)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Active configuration is supported.

Table 35 describes class CIM_ElementCapabilities (iSCSIConfigurationCapabilities to iSCSIConfigurationService).

Table 35 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (iSCSIConfigurationCapabilities to iSCSIConfigurationService)

Properties	Requirement	Description and Notes
Capabilities	Mandatory	
ManagedElement	Mandatory	

10.8.9 CIM_ElementSettingData (iSCSIConnectionSettings to TCPProtocolEndpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 36 describes class CIM_ElementSettingData (iSCSIConnectionSettings to TCPProtocolEndpoint).

Table 36 - SMI Referenced Properties/Methods for CIM_ElementSettingData (iSCSIConnectionSettings to TCPProtocolEndpoint)

Properties	Requirement	Description and Notes
SettingData	Mandatory	
ManagedElement	Mandatory	

10.8.10 CIM_ElementSettingData (iSCSIConnectionSettings to iSCSIProtocolEndpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 37 describes class CIM_ElementSettingData (iSCSIConnectionSettings to iSCSIProtocolEndpoint).

Table 37 - SMI Referenced Properties/Methods for CIM_ElementSettingData (iSCSIConnectionSettings to iSCSIProtocolEndpoint)

Properties	Requirement	Description and Notes
SettingData	Mandatory	
ManagedElement	Mandatory	

10.8.11 CIM_ElementSettingData (iSCSIConnectionSettings to SCSIProtocolController)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 38 describes class CIM_ElementSettingData (iSCSIConnectionSettings to SCSIProtocolController).

Table 38 - SMI Referenced Properties/Methods for CIM_ElementSettingData (iSCSIConnectionSettings to SCSIProtocolController)

Properties	Requirement	Description and Notes
SettingData	Mandatory	
ManagedElement	Mandatory	

10.8.12 CIM_ElementSettingData (iSCSIConnectionSettings to System)

Created By: Static

Modified By: Static
Deleted By: Static
Requirement: Mandatory

Table 39 describes class CIM_ElementSettingData (iSCSI Session Settings to System).

Table 39 - SMI Referenced Properties/Methods for CIM_ElementSettingData (iSCSI Session Settings to System)

Properties	Requirement	Description and Notes
ManagedElement	Mandatory	
SettingData	Mandatory	

10.8.13 CIM_ElementSettingData (iSCSI Session Settings to iSCSI Protocol Endpoint)

Created By: Static
Modified By: Static
Deleted By: Static
Requirement: Mandatory

Table 40 describes class CIM_ElementSettingData (iSCSI Session Settings to iSCSI Protocol Endpoint).

Table 40 - SMI Referenced Properties/Methods for CIM_ElementSettingData (iSCSI Session Settings to iSCSI Protocol Endpoint)

Properties	Requirement	Description and Notes
SettingData	Mandatory	
ManagedElement	Mandatory	

10.8.14 CIM_ElementStatisticalData (iSCSI Login Statistics to SCSI Protocol Controller)

Created By: Static
Modified By: Static
Deleted By: Static
Requirement: Mandatory

Table 41 describes class CIM_ElementStatisticalData (iSCSI Login Statistics to SCSI Protocol Controller).

Table 41 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (iSCSI Login Statistics to SCSI Protocol Controller)

Properties	Requirement	Description and Notes
Stats	Mandatory	
ManagedElement	Mandatory	

10.8.15 CIM_ElementStatisticalData (iSCSI Session Failures to SCSI Protocol Controller)

Created By: Static
 Modified By: Static
 Deleted By: Static
 Requirement: Mandatory

Table 42 describes class CIM_ElementStatisticalData (iSCSI_SessionFailures to SCSIProtocolController).

Table 42 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (iSCSI_SessionFailures to SCSIProtocolController)

Properties	Requirement	Description and Notes
Stats	Mandatory	
ManagedElement	Mandatory	

10.8.16 CIM_ElementStatisticalData (iSCSI_SessionStatistics to iSCSI_Session)

Created By: Static
 Modified By: Static
 Deleted By: Static
 Requirement: Mandatory

Table 43 describes class CIM_ElementStatisticalData (iSCSI_SessionStatistics to iSCSI_Session).

Table 43 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (iSCSI_SessionStatistics to iSCSI_Session)

Properties	Requirement	Description and Notes
Stats	Mandatory	
ManagedElement	Mandatory	

10.8.17 CIM_EndpointOfNetworkPipe (iSCSI_Connection to TCPProtocolEndpoint)

Created By: Static
 Modified By: Static
 Deleted By: Static
 Requirement: Mandatory

Table 44 describes class CIM_EndpointOfNetworkPipe (iSCSI_Connection to TCPProtocolEndpoint).

Table 44 - SMI Referenced Properties/Methods for CIM_EndpointOfNetworkPipe (iSCSI_Connection to TCPProtocolEndpoint)

Properties	Requirement	Description and Notes
Dependent	Mandatory	
Antecedent	Mandatory	

10.8.18 CIM_EndpointOfNetworkPipe (iSCSI Session to iSCSI Protocol Endpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 45 describes class CIM_EndpointOfNetworkPipe (iSCSI Session to iSCSI Protocol Endpoint).

Table 45 - SMI Referenced Properties/Methods for CIM_EndpointOfNetworkPipe (iSCSI Session to iSCSI Protocol Endpoint)

Properties	Requirement	Description and Notes
Dependent	Mandatory	
Antecedent	Mandatory	

10.8.19 CIM_EthernetPort

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 46 describes class CIM_EthernetPort.

Table 46 - SMI Referenced Properties/Methods for CIM_EthernetPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus	Mandatory	
PermanentAddress	Mandatory	

10.8.20 CIM_HostedAccessPoint (System to IP Protocol Endpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 47 describes class CIM_HostedAccessPoint (System to IPProtocolEndpoint).

Table 47 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (System to IPProtocolEndpoint)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

10.8.21 CIM_HostedAccessPoint (System to TCPProtocolEndpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 48 describes class CIM_HostedAccessPoint (System to TCPProtocolEndpoint).

Table 48 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (System to TCPProtocolEndpoint)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

10.8.22 CIM_HostedAccessPoint (System to iSCSIProtocolEndpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 49 describes class CIM_HostedAccessPoint (System to iSCSIProtocolEndpoint).

Table 49 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (System to iSCSIProtocolEndpoint)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

10.8.23 CIM_HostedCollection

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 50 describes class CIM_HostedCollection.

Table 50 - SMI Referenced Properties/Methods for CIM_HostedCollection

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

10.8.24 CIM_HostedService

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 51 describes class CIM_HostedService.

Table 51 - SMI Referenced Properties/Methods for CIM_HostedService

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

10.8.25 CIM_IPProtocolEndpoint

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 52 describes class CIM_IPProtocolEndpoint.

Table 52 - SMI Referenced Properties/Methods for CIM_IPProtocolEndpoint

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
IPv4Address	Optional	
IPv6Address	Optional	
ProtocolIFType	Mandatory	

10.8.26 CIM_MemberOfCollection

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 53 describes class CIM_MemberOfCollection.

Table 53 - SMI Referenced Properties/Methods for CIM_MemberOfCollection

Properties	Requirement	Description and Notes
Member	Mandatory	
Collection	Mandatory	

10.8.27 CIM_NetworkPipeComposition

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 54 describes class CIM_NetworkPipeComposition.

Table 54 - SMI Referenced Properties/Methods for CIM_NetworkPipeComposition

Properties	Requirement	Description and Notes
PartComponent	Mandatory	
GroupComponent	Mandatory	

10.8.28 CIM_SAPAvailableForElement

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 55 describes class CIM_SAPAvailableForElement.

Table 55 - SMI Referenced Properties/Methods for CIM_SAPAvailableForElement

Properties	Requirement	Description and Notes
ManagedElement	Mandatory	
AvailableSAP	Mandatory	

10.8.29 CIM_SCSIProtocolController

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 56 describes class CIM_SCSIProtocolController.

Table 56 - SMI Referenced Properties/Methods for CIM_SCSIProtocolController

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
ElementName	Mandatory	iSCSI Alias.
Name	Mandatory	
NameFormat	Mandatory	

10.8.30 CIM_SystemDevice (System to EthernetPort)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 57 describes class CIM_SystemDevice (System to EthernetPort).

Table 57 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to EthernetPort)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	
PartComponent	Mandatory	

10.8.31 CIM_SystemDevice (System to SCSIProtocolController)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 58 describes class CIM_SystemDevice (System to SCSIProtocolController).

Table 58 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to SCSIProtocol-Controller)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	
PartComponent	Mandatory	

10.8.32 CIM_SystemSpecificCollection

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 59 describes class CIM_SystemSpecificCollection.

Table 59 - SMI Referenced Properties/Methods for CIM_SystemSpecificCollection

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
ElementName	Mandatory	

10.8.33 CIM_TCPProtocolEndpoint

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 60 describes class CIM_TCPProtocolEndpoint.

Table 60 - SMI Referenced Properties/Methods for CIM_TCPProtocolEndpoint

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
PortNumber	Mandatory	
ProtocolIFType	Mandatory	

10.8.34 CIM_iSCSICapabilities

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 61 describes class CIM_iSCSICapabilities.

Table 61 - SMI Referenced Properties/Methods for CIM_iSCSICapabilities

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
ElementName	Mandatory	

Table 61 - SMI Referenced Properties/Methods for CIM_iSCSICapabilities

Properties	Requirement	Description and Notes
MinimumSpecificationVersionSupported	Mandatory	
MaximumSpecificationVersionSupported	Mandatory	
AuthenticationMethodsSupported	Mandatory	

10.8.35 CIM_iSCSIConfigurationCapabilities

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Active configuration is supported.

Table 62 describes class CIM_iSCSIConfigurationCapabilities.

Table 62 - SMI Referenced Properties/Methods for CIM_iSCSIConfigurationCapabilities

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
ElementName	Mandatory	
iSCSINodeCreationSupported	Mandatory	
iSCSIProtocolEndpointCreationSupported	Mandatory	
IdentifierSelectionSupported	Mandatory	

10.8.36 CIM_iSCSIConfigurationService

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 63 describes class CIM_iSCSIConfigurationService.

Table 63 - SMI Referenced Properties/Methods for CIM_iSCSIConfigurationService

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	

10.8.37 CIM_iSCSIConnection

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 64 describes class CIM_iSCSIConnection.

Table 64 - SMI Referenced Properties/Methods for CIM_iSCSIConnection

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
ConnectionID	Mandatory	
MaxReceiveDataSegmentLength	Mandatory	
MaxTransmitDataSegmentLength	Mandatory	
HeaderDigestMethod	Mandatory	
OtherHeaderDigestMethod	Optional	
DataDigestMethod	Mandatory	
OtherDataDigestMethod	Optional	
ReceivingMarkers	Mandatory	
SendingMarkers	Mandatory	
ActiveiSCSIVersion	Mandatory	
AuthenticationMethodUsed	Mandatory	
MutualAuthentication	Mandatory	

10.8.38 CIM_iSCSIConnectionSettings

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 65 describes class CIM_iSCSIConnectionSettings.

Table 65 - SMI Referenced Properties/Methods for CIM_iSCSIConnectionSettings

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
ElementName	Mandatory	
MaxReceiveDataSegmentLength	Mandatory	
PrimaryHeaderDigestMethod	Mandatory	
OtherPrimaryHeaderDigestMethod	Optional	
PrimaryDataDigestMethod	Mandatory	
OtherPrimaryDataDigestMethod	Optional	
SecondaryHeaderDigestMethod	Mandatory	
OtherSecondaryHeaderDigestMethod	Optional	
SecondaryDataDigestMethod	Mandatory	
OtherSecondaryDataDigestMethod	Optional	

Table 65 - SMI Referenced Properties/Methods for CIM_iSCSIConnectionSettings

Properties	Requirement	Description and Notes
RequestingMarkersOnReceive	Mandatory	
PrimaryAuthenticationMethod	Mandatory	
SecondaryAuthenticationMethod	Mandatory	

10.8.39 CIM_iSCSILoginStatistics

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 66 describes class CIM_iSCSILoginStatistics.

Table 66 - SMI Referenced Properties/Methods for CIM_iSCSILoginStatistics

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
ElementName	Mandatory	
LoginFailures	Optional	
LastLoginFailureTime	Optional	
LastLoginFailureType	Optional	
OtherLastLoginFailureType	Optional	
LastLoginFailureRemoteNodeName	Optional	
LastLoginFailureRemoteAddressType	Optional	
LastLoginFailureRemoteAddress	Optional	
SuccessfulLogins	Optional	
NegotiationLoginFailures	Optional	
AuthenticationLoginFailures	Optional	
AuthorizationLoginFailures	Optional	
LoginRedirects	Optional	
OtherLoginFailures	Optional	
NormalLogouts	Optional	
OtherLogouts	Optional	

10.8.40 CIM_iSCSIProtocolEndpoint

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 67 describes class CIM_iSCSIProtocolEndpoint.

Table 67 - SMI Referenced Properties/Methods for CIM_iSCSIProtocolEndpoint

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ConnectionType	Mandatory	iSCSI.
Identifier	Mandatory	ISID or TPGT.
ProtocolIFType	Mandatory	Other.
OtherTypeDescription	Mandatory	
Role	Mandatory	Shall be 3 (Target) or 4 (Both Initiator and Target).

10.8.41 CIM_iSCSISession

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 68 describes class CIM_iSCSISession.

Table 68 - SMI Referenced Properties/Methods for CIM_iSCSISession

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
Directionality	Mandatory	
SessionType	Mandatory	
TSIH	Mandatory	
EndPointName	Mandatory	
CurrentConnections	Mandatory	
InitialR2T	Mandatory	
ImmediateData	Mandatory	
MaxOutstandingR2T	Mandatory	
MaxUnsolicitedFirstDataBurstLength	Mandatory	
MaxDataBurstLength	Mandatory	
DataSequenceInOrder	Mandatory	
DataPDUInOrder	Mandatory	
ErrorRecoveryLevel	Mandatory	
MaxConnectionsPerSession	Mandatory	
DefaultTimeToWait	Mandatory	
DefaultTimeToRetain	Mandatory	

10.8.42 CIM_iSCSISessionFailures

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 69 describes class CIM_iSCSISessionFailures.

Table 69 - SMI Referenced Properties/Methods for CIM_iSCSISessionFailures

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
ElementName	Mandatory	
SessionFailures	Optional	
LastSessionFailureType	Optional	
OtherLastSessionFailureType	Optional	
LastSessionFailureRemoteNodeName	Optional	
SessionDigestFailures	Optional	
SessionConnectionTimeoutFailures	Optional	
SessionFormatErrors	Optional	

10.8.43 CIM_iSCSISessionSettings

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 70 describes class CIM_iSCSISessionSettings.

Table 70 - SMI Referenced Properties/Methods for CIM_iSCSISessionSettings

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
ElementName	Mandatory	
MaxConnectionsPerSession	Mandatory	
InitialR2TPreference	Mandatory	
ImmediateDataPreference	Mandatory	
MaxOutstandingR2T	Mandatory	
MaxUnsolicitedFirstDataBurstLength	Mandatory	
MaxDataBurstLength	Mandatory	
DataSequenceInOrderPreference	Mandatory	
DataPDUInOrderPreference	Mandatory	

Table 70 - SMI Referenced Properties/Methods for CIM_iSCSISessionSettings

Properties	Requirement	Description and Notes
DefaultTimeToWaitPreference	Mandatory	
DefaultTimeToRetainPreference	Mandatory	
ErrorRecoveryLevelPreference	Mandatory	

10.8.44 CIM_iSCSISessionStatistics

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 71 describes class CIM_iSCSISessionStatistics.

Table 71 - SMI Referenced Properties/Methods for CIM_iSCSISessionStatistics

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
ElementName	Mandatory	
CommandPDUsTransferred	Optional	
ResponsePDUsTransferred	Optional	
BytesTransmitted	Optional	
BytesReceived	Optional	
DigestErrors	Optional	
ConnectionTimeoutErrors	Optional	

STABLE

EXPERIMENTAL

11 Serial Attached SCSI (SAS) Target Port Subprofile

11.1 Synopsis

Profile Name: SAS Target Ports (Component Profile)

Version: 1.4.0

Organization: SNIA

CIM Schema Version: 2.13.1

Table 72 describes the related profiles for SAS Target Ports.

Table 72 - Related Profiles for SAS Target Ports

Profile Name	Organization	Version	Requirement	Description
Indication	SNIA	1.5.0	Mandatory	

Central Class: CIM_SASPort

Scoping Class: a CIM_System in a separate autonomous profile

11.2 Description

11.2.1 General

Figure 11 illustrates the Serial Attached SCSI (SAS) Target Port. Serial Attached SCSI is a lower cost network interface for SCSI communication.

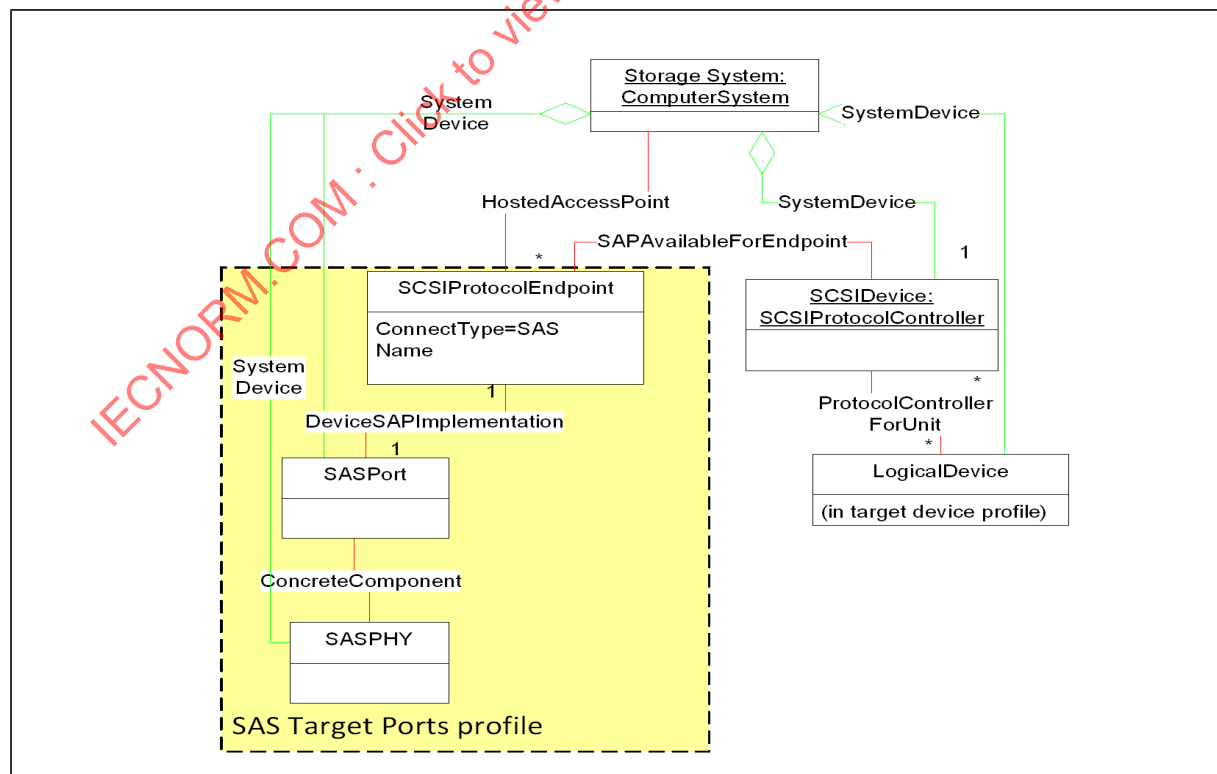


Figure 11 - Serial Attached SCSI (SAS) Target Port Instance Diagram

SCSIProtocolEndpoint.ConnectionType shall be set to "SAS". SASPort represents the port and is connected to SCSIProtocolEndpoint by DeviceSAPImplementation. The SASPort contains information about the speed for the bus.

The SASPHY class represents a SAS PHY. A SAS Port may have multiple associated PHYs; generally, all the PHYs are connected to the same initiator or expander and provide additional bandwidth.

11.2.2 Health and Fault Management

Table 73 describes SASPort OperationalStatus.

Table 73 - SASPort OperationalStatus

OperationalStatus	Description
OK	Port is online
Error	Port has a failure
Stopped	Port is disabled
InService	Port is in Self Test
Unknown	

11.3 Methods

11.3.1 Extrinsic Methods of this Subprofile

None.

11.3.2 Intrinsic Methods of this Subprofile

The profile supports read methods and association traversal. Specifically, the list of intrinsic operations supported are as follows:

- GetInstance
- Associators
- AssociatorNames
- References
- ReferenceNames
- EnumerateInstances
- EnumerateInstanceNames

11.4 Client considerations and Recipes

None.

11.5 CIM Elements

11.5.1 General

Table 74 describes the CIM elements for SAS Target Ports.

Table 74 - CIM Elements for SAS Target Ports

Element Name	Requirement	Description
11.5.2 CIM_ConcreteComponent	Mandatory	Associates SASPort and SASPHY.
11.5.3 CIM_DeviceSAPImplementation	Mandatory	Associates SASPort and SCSIProtocolEndpoint.
11.5.4 CIM_HostedAccessPoint	Mandatory	Associates ComputerSystem to ProtocolEndpoint.
11.5.5 CIM_SASPort	Mandatory	Represents the logical aspects of the physical port and may have multiple associated protocols.
11.5.6 CIM_SCSIProtocolEndpoint	Mandatory	Represents management characteristics related to the SCSI command set.
11.5.7 CIM_SystemDevice (Port)	Mandatory	Associates ComputerSystem to LogicalPort.
11.5.8 CIM_SystemDevice (SAS PHY)	Mandatory	Associates ComputerSystem to SASPHY.
11.5.9 SNIA_SASPHY	Mandatory	Several SASPHYs may together be aggregated into a SAS Logical Port.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_SASPort	Mandatory	CQL -Create SASPort.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_SASPort AND SourceInstance.CIM_SASPort::OperationalStatus <> PreviousInstance.SAS_Port::OperationalStatus	Mandatory	CQL -Modify SASPort.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_SASPort	Mandatory	CQL -Delete SASPort.

11.5.2 CIM_ConcreteComponent

Associates SASPort and SASPHY.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 75 describes class CIM_ConcreteComponent.

Table 75 - SMI Referenced Properties/Methods for CIM_ConcreteComponent

Properties	Requirement	Description and Notes
PartComponent	Mandatory	Reference to SASPHY.
GroupComponent	Mandatory	Reference to SASPort.

11.5.3 CIM_DeviceSAPImplementation

Associates SASPort and SCSIProtocolEndpoint. The class definition specializes the CIM_DeviceSAPImplementation definition in the Generic Target Ports profile. Properties or

methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 76 describes class CIM_DeviceSAPImplementation.

Table 76 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Requirement	Description and Notes
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint.
Antecedent (overridden)	Mandatory	Reference to SASPort.

11.5.4 CIM_HostedAccessPoint

Associates ComputerSystem to ProtocolEndpoint. Limit to targets. The class definition specializes the CIM_HostedAccessPoint definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 77 describes class CIM_HostedAccessPoint.

Table 77 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in the referencing profile.
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint.

11.5.5 CIM_SASPort

Represents the logical aspects of the physical port and may have multiple associated protocols. The class definition specializes the CIM_LogicalPort definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 78 describes class CIM_SASPort.

Table 78 - SMI Referenced Properties/Methods for CIM_SASPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus	Mandatory	
UsageRestriction (overridden)	Mandatory	Shall be 2 (Front-end Only).
PortType (overridden)	Mandatory	Shall be 94 (SAS).
PermanentAddress (added)	Mandatory	SAS Address. Shall be 16 un-separated upper case hex digits.

11.5.6 CIM_SCSIProtocolEndpoint

Represents management characteristics related to the SCSI command set. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 79 describes class CIM_SCSIProtocolEndpoint.

Table 79 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType	Mandatory	Shall be 1 (Other).
OtherTypeDescription (overridden)	Mandatory	Shall be the string 'SCSI'.
ConnectionType (added)	Mandatory	Shall be 8 (SAS).
Role (added)	Mandatory	Shall be 3 (Target).

11.5.7 CIM_SystemDevice (Port)

Associates ComputerSystem to LogicalPort. The class definition specializes the CIM_SystemDevice definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 80 describes class CIM_SystemDevice (Port).

Table 80 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to ComputerSystem in the referencing profile.
PartComponent (overridden)	Mandatory	Reference to SASPort.

11.5.8 CIM_SystemDevice (SAS PHY)

Associates ComputerSystem to SASPHY.

Requirement: Mandatory

Table 81 describes class CIM_SystemDevice (SAS PHY).

Table 81 - SMI Referenced Properties/Methods for CIM_SystemDevice (SAS PHY)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to ComputerSystem.
PartComponent	Mandatory	Reference to SASPHY.

11.5.9 SNIA_SASPHY

Several SASPHYs may together be aggregated into a SAS Logical Port.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 82 describes class SNIA_SASPHY.

Table 82 - SMI Referenced Properties/Methods for SNIA_SASPHY

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
HardwareMinimumPhysicalLinkRate	Mandatory	
HardwareMaximumPhysicalLinkRate	Mandatory	
ProgrammedMinimumPhysicalLinkRate	Mandatory	
ProgrammedMaximumPhysicalLinkRate	Mandatory	
NegotiatedPhysicalLinkRate	Mandatory	

EXPERIMENTAL

EXPERIMENTAL

12 Serial ATA (SATA) Target Ports Profile

12.1 Synopsis

Profile Name: SATA Target Ports (Component Profile)

Version: 1.4.0

Organization: SNIA

CIM Schema Version: 2.13.1

Table 83 describes the related profiles for SATA Target Ports.

Table 83 - Related Profiles for SATA Target Ports

Profile Name	Organization	Version	Requirement	Description
Indication	SNIA	1.5.0	Mandatory	

Central Class: CIM_SASPortt

Scoping Class: a CIM_System in a separate autonomous profile

Model Serial ATA (SATA) target ports.

12.2 Description

12.2.1 General

Figure 12 illustrates the Serial ATA Target Port Profile. Serial ATA has a simple bus structure. The SATAPort class will include attributes that specifies bus speed and other hardware options.

This model will not be used with LMM common subprofile. All nodes on the bus will have access to each other.

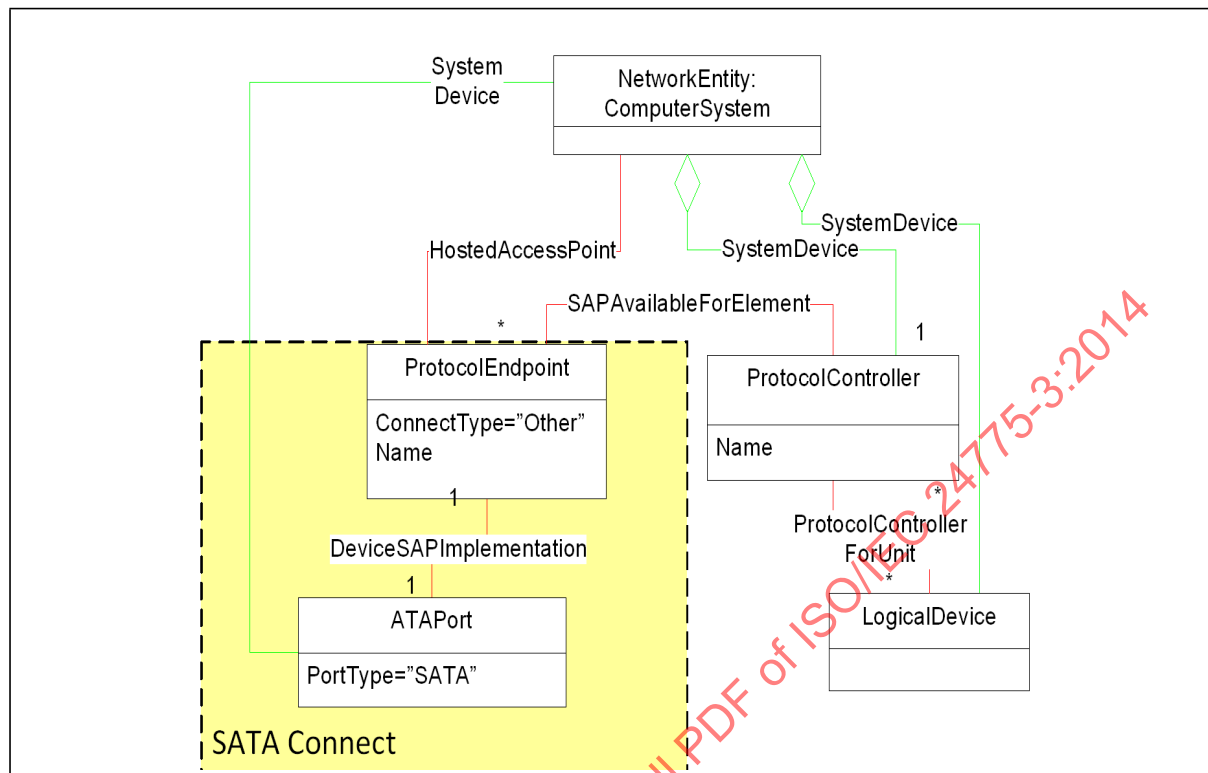


Figure 12 - SATA Target Port Instance Diagram

ProtocolEndpoint.ConnectionType shall be set to "other". The ProtocolEndPoint class is associated to the ATAPort class with DevImplementation. The ATAPort class contains all the bus operational settings.

12.2.2 Health and Fault Management

Table 84 describes ATAPort OperationalStatus.

Table 84 - ATAPort OperationalStatus

OperationalStatus	Description
OK	Port is online
Error	Port has a failure
Stopped	Port is disabled
InService	Port is in Self Test
Unknown	

12.3 Methods of this Subprofile

None

12.4 Client considerations and Recipes

None

12.5 CIM elements

12.5.1 General

Table 85 describes the CIM elements for SATA Target Ports.

Table 85 - CIM Elements for SATA Target Ports

Element Name	Requirement	Description
12.5.2 CIM_ATAPort	Mandatory	Represents the logical aspects of the physical port and may have multiple associated protocols.
12.5.3 CIM_ATAProtocolEndpoint	Mandatory	Represents management characteristics related to the ATA command set.
12.5.4 CIM_DeviceSAPImplementation	Mandatory	Associates front-end ATAPort and target ATAProtocolEndpoint.
12.5.5 CIM_HostedAccessPoint	Mandatory	Associates ComputerSystem to ATAProtocolEndpoint.
12.5.6 CIM_SystemDevice (Port)	Mandatory	Associates ComputerSystem to ATAPort.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_ATAPort	Mandatory	Create ATAPort.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ATAPort AND SourceInstance.OperationalStatus <> PreviousInstance.OperationalStatus	Mandatory	Modify ATAPort.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_ATAPort	Mandatory	Delete ATAPort.

12.5.2 CIM_ATAPort

Represents the logical aspects of the physical port and may have multiple associated protocols. The class definition specializes the CIM_LogicalPort definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 86 describes class CIM_ATAPort.

Table 86 - SMI Referenced Properties/Methods for CIM_ATAPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus	Mandatory	
UsageRestriction (overridden)	Mandatory	Shall be 2 (Front-end Only).
PortType (overridden)	Mandatory	Shall be 92 93 (SATA or SATA2).

12.5.3 CIM_ATAProtocolEndpoint

Represents management characteristics related to the ATA command set. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 87 describes class CIM_ATAProtocolEndpoint.

Table 87 - SMI Referenced Properties/Methods for CIM_ATAProtocolEndpoint

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType	Mandatory	Shall be 1 (Other).
OtherTypeDescription (overridden)	Mandatory	Shall be the string 'ATA'.
ConnectionType (added)	Mandatory	Shall be 3 (SATA).
Role (added)	Mandatory	Shall be 3 (Target).

12.5.4 CIM_DeviceSAPImplementation

Associates front-end ATAPort and target ATAProtocolEndpoint. Limit to target ProtocolEndpoints and front-end ports. The class definition specializes the CIM_DeviceSAPImplementation definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 88 describes class CIM_DeviceSAPImplementation.

Table 88 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Requirement	Description and Notes
Dependent (overridden)	Mandatory	Reference to ATAProtocolEndpoint.
Antecedent (overridden)	Mandatory	Reference to ATAPort.

12.5.5 CIM_HostedAccessPoint

Associates ComputerSystem to ATAProtocolEndpoint. Limit to targets. The class definition specializes the CIM_HostedAccessPoint definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static
 Modified By: Static
 Deleted By: Static
 Requirement: Mandatory

Table 89 describes class CIM_HostedAccessPoint.

Table 89 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint

Properties	Requirement	Description and Notes
Antecedent (overridden)	Mandatory	
Dependent (overridden)	Mandatory	Reference to ATAProtocolEndpoint.

12.5.6 CIM_SystemDevice (Port)

Associates ComputerSystem to ATAPort. The class definition specializes the CIM_SystemDevice definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static
 Modified By: Static
 Deleted By: Static
 Requirement: Mandatory

Table 90 describes class CIM_SystemDevice (Port).

Table 90 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port)

Properties	Requirement	Description and Notes
GroupComponent (overridden)	Mandatory	Reference to ComputerSystem in referencing profile.
PartComponent (overridden)	Mandatory	Reference to ATAPort.

EXPERIMENTAL

EXPERIMENTAL

13 SB Target Ports Profile

13.1 Synopsis

Profile Name: SB Target Ports (Component Profile)

Version: 1.2.0

Organization: SNIA

CIM Schema Version: 2.13

Table 91 describes the related profiles for SB Target Ports.

Table 91 - Related Profiles for SB Target Ports

Profile Name	Organization	Version	Requirement	Description
Indication	SNIA	1.5.0	Mandatory	

Central Class: CIM_FCPort

Scoping Class: CIM_System

13.2 Description

The SB Target Port Profile models the SB (Single Byte) Fibre Channel specific aspects of a target storage system. The Single Byte protocols are FC4 protocols that support mainframe IO (as opposed to SCSI, which supports IO from non-mainframe systems such as Unix or Windows systems).

The SB Target Port Profile provides a way for storage profiles to model target ports that are dedicated to serving SB hosts attachment. With this support a client will be able to distinguish FC ports that are provided for SCSI access from FC Ports that are provided for mainframe attachment. This is an important distinction for management, since fabric connectivity collections for SB would typically be separate for fabric connectivity collections for SCSI. Similarly, management functions for masking and mapping are somewhat different for SB than SCSI. So, it is important for management applications to be aware of the distinctions.

The SB Target Port Profile specializes the Generic Target Port Profile.

For SB enabled Fibre Channel ports, the concrete subclass of LogicalPort is FCPort. FCPort is always associated 1-1 with a SNIA_SBProtocolEndpoint instance.

13.3 Implementation

Figure 13 illustrates the SB Target Port Profile.

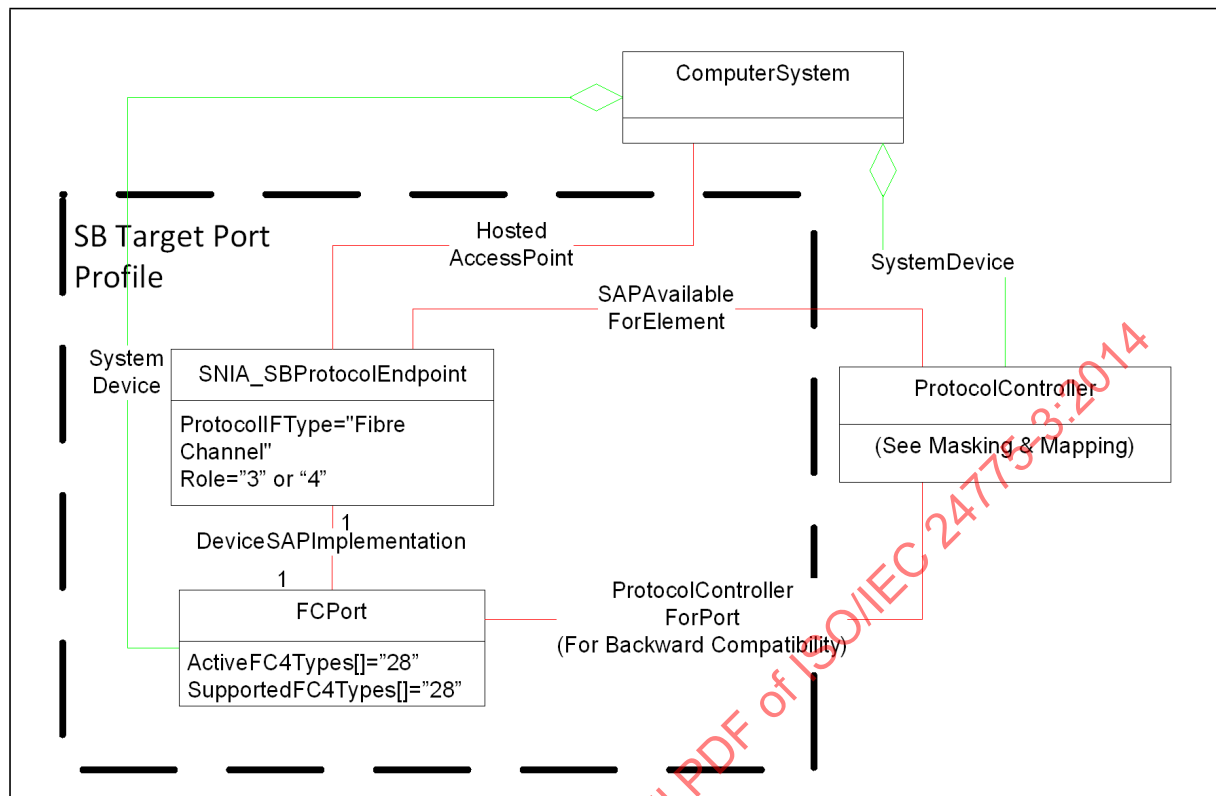


Figure 13 - SB Target Port Instance Diagram

SB Ports are Fibre Channel Ports with the SupportedFC4Types[] and ActiveFC4Types[] arrays holding the value "28" (for "FC-SB-2 Control Unit"). The SupportedFC4Types[] property shall contain the value "28". The ActiveFC4Types[] property shall contain the value "28" for FCPorts that are actively supporting SB protocols.

The FCPort shall also support an SBProtocolEndpoint with a role property of either "3" ("Target") or "4" ("Both initiator and target").

For the SB Target Port Profile, the FCPort is the central class of the Profile.

13.4 Health and Fault Management Consideration

Table 92 defines the SMI-S defined meanings of the OperationalStatus property for FCPorts used in the SB Target Port Profile.

Table 92 - FCPort OperationalStatus

OperationalStatus	Description
OK	Port is online
Error	Port has a failure
Stopped	Port is disabled
InService	Port is in self test
Unknown	

13.5 Cascading Considerations

None

13.6 Methods of the Profile

13.6.1 Extrinsic Methods of the Profile

None

13.6.2 Intrinsic Methods of the Profile

The profile supports read methods and association traversal. Specifically, the list of intrinsic operations supported are as follows:

- GetInstance
- Associators
- AssociatorNames
- References
- ReferenceNames
- EnumerateInstances
- EnumerateInstanceNames

13.7 Client Considerations and Recipes

None

13.8 CIM Elements

13.8.1 General

Table 93 describes the CIM elements for SB Target Ports.

Table 93 - CIM Elements for SB Target Ports

Element Name	Requirement	Description
13.8.2 CIM_DeviceSAPImplementation	Mandatory	Associates front-end FCPort and target SBProtocolEndpoint.
13.8.3 CIM_FCPort	Mandatory	The FCPort provides transport for SB protocols.
13.8.4 CIM_HostedAccessPoint	Mandatory	Associates ComputerSystem to SBProtocolEndpoint.
13.8.5 CIM_SystemDevice (Port)	Mandatory	Associates controller ComputerSystem to FCPort.
13.8.6 SNIA_SBProtocolEndpoint	Mandatory	This defines the protocol being used on the FC Port as SB (Single Byte).
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_FCPort	Mandatory	Create FCPort.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.OperationalStatus <> PreviousInstance.OperationalStatus	Mandatory	Deprecated WQL -Change to FCPort OperationalStatus.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.CIM_FCPort::OperationalStatus <> PreviousInstance.CIM_FCPort::OperationalStatus	Mandatory	CQL -Change to FCPort OperationalStatus.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_FCPort	Mandatory	Delete FCPort.

13.8.2 CIM_DeviceSAPImplementation

Associates front-end FCPort and target SBProtocolEndpoint. Limit to target ProtocolEndpoints and front-end ports. The class definition specializes the CIM_DeviceSAPImplementation definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 94 describes class CIM_DeviceSAPImplementation.

Table 94 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Requirement	Description and Notes
Dependent (overridden)	Mandatory	
Antecedent (overridden)	Mandatory	

13.8.3 CIM_FCPort

The CIM_FCPort class for SB Target Ports is the same as the FC Target Port class, except that the SupportedFC4Types and ActiveFC4Types properties are mandatory, and the SupportedFC4Types shall contain "28" and the ActiveFC4Types should contain "28" to indicate support for FC-SB-2 Control Unit functions. The class definition specializes the CIM_LogicalPort definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 95 describes class CIM_FCPort.

Table 95 - SMI Referenced Properties/Methods for CIM_FCPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus	Mandatory	
UsageRestriction (overridden)	Mandatory	Shall be 2 (Front-end Only).
PortType (overridden)	Optional	
PermanentAddress (added)	Mandatory	Port WWN. Shall be 16 unseparated uppercase hex digits.
SupportedCOS (added)	Optional	

Table 95 - SMI Referenced Properties/Methods for CIM_FCPort

Properties	Requirement	Description and Notes
ActiveCOS (added)	Optional	
SupportedFC4Types (added)	Mandatory	For SB Target Ports this array shall contain 28 (FC-SB-2 Control Unit).
ActiveFC4Types (added)	Mandatory	For SB Target Ports this array should contain 28 (FC-SB-2 Control Unit).

13.8.4 CIM_HostedAccessPoint

Associates ComputerSystem to SBProtocolEndpoint. Limit to targets (Role = 3). The class definition specializes the CIM_HostedAccessPoint definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 96 describes class CIM_HostedAccessPoint.

Table 96 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in the referencing profile.
Dependent (overridden)	Mandatory	Reference to SBProtocolEndpoint.

13.8.5 CIM_SystemDevice (Port)

Associates controller ComputerSystem to FCPort. The class definition specializes the CIM_SystemDevice definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 97 describes class CIM_SystemDevice (Port).

Table 97 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port)

Properties	Requirement	Description and Notes
GroupComponent (overridden)	Mandatory	Reference to ComputerSystem in the referencing profile.
PartComponent (overridden)	Mandatory	Reference to FCPort.

13.8.6 SNIA_SBProtocolEndpoint

The SNIA_SBProtocolEndpoint specializes the Generic Target Port CIM_ProtocolEndpoint. The main difference is that the OtherTypeDescription is set to "SB". The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Target Ports profile. Properties

or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 98 describes class SNIA_SBProtocolEndpoint.

Table 98 - SMI Referenced Properties/Methods for SNIA_SBProtocolEndpoint

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType	Mandatory	Shall be 1 (Other).
OtherTypeDescription (overridden)	Mandatory	Shall be the string 'SB'.
Role (added)	Mandatory	Shall be 3 (Target).
ConnectionType (added)	Mandatory	Shall be 2 (Fibre Channel).

EXPERIMENTAL

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

EXPERIMENTAL

14 Direct Attach (DA) Ports Profile

14.1 Description

The DAPort (Direct Attach) port models storage systems that attach directly to buses in a host system (e.g., PCI, PCI-E, and chip interfaces on a motherboard). The DAPort can be viewed as both the initiator and Target ports.

This port can not be used with the LUN Mapping and Masking Profile. All volumes served by this port are fully accessible by the host system.

Figure 14 illustrates the Direct Attach (DA) Ports Profile. Volumes served by this port shall be discovered and presented by the Host Discovered Resources Profile.

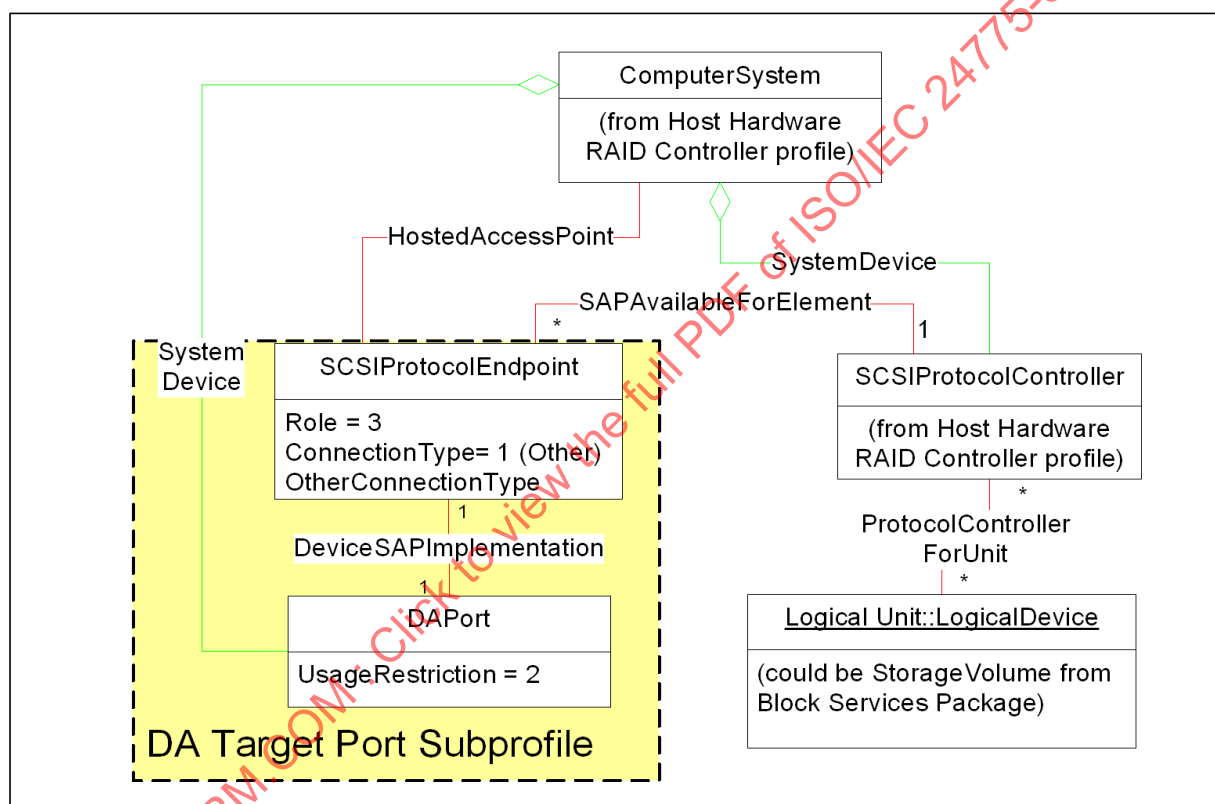


Figure 14 - DA Port Instance Diagram

The DAPort class is connected to the ProtocolEndpoint using the DeviceSAPImplementation association and to the controller ComputerSystem (in the Host Hardware RAID Controller Profile) using SystemDevice. SCSIProtocolController is associated to the controller ComputerSystem from the same ComputerSystem using HostedAccessPoint.

14.2 Health and Fault Management

Not defined in this International Standard.

14.3 Supported Profiles and Packages

Related Profiles for DA Target Ports: Not defined in this International Standard.

14.4 Extrinsic Methods

Not defined in this International Standard.

14.5 Client Considerations and Recipes

Not defined in this International Standard.

14.6 Registered Name and Version

DA Target Ports version 1.4.0 (Component Profile)

Specializes SNIA Generic Target Ports version 1.4.0

14.7 CIM Elements

14.7.1 General

Table 99 describes the CIM elements for DA Target Ports.

Table 99 - CIM Elements for DA Target Ports

Element Name	Requirement	Description
14.7.2 CIM_DAPort	Mandatory	Models the "port" emulated by a host RAID controller.
14.7.3 CIM_DeviceSAPImplementation	Mandatory	Associates DAPort and SCSIProtocolEndpoint.
14.7.4 CIM_HostedAccessPoint	Mandatory	Associates ComputerSystem to SCSIProtocolEndpoint.
14.7.5 CIM_SCSIProtocolEndpoint	Mandatory	ProtocolEndpoint Associates representing support for SCSI.
14.7.6 CIM_SystemDevice (Port)	Mandatory	Associates controller ComputerSystem to DAPort.

14.7.2 CIM_DAPort

Models the "port" emulated by a host RAID controller. The class definition specializes the CIM_LogicalPort definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 100 describes class CIM_DAPort.

Table 100 - SMI Referenced Properties/Methods for CIM_DAPort

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
DeviceID		Mandatory	

Table 100 - SMI Referenced Properties/Methods for CIM_DAPort

Properties	Flags	Requirement	Description and Notes
OperationalStatus		Mandatory	
UsageRestriction (overridden)		Mandatory	Shall be 2 (Front-end Only).
PortType (overridden)		Optional	Not defined in this version of the International Standard.

14.7.3 CIM_DeviceSAPImplementation

Associates DAPort and SCSIProtocolEndpoint. The class definition specializes the CIM_DeviceSAPImplementation definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 101 describes class CIM_DeviceSAPImplementation.

Table 101 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Flags	Requirement	Description and Notes
Dependent (overridden)		Mandatory	Reference to SCSIProtocolEndpoint.
Antecedent (overridden)		Mandatory	Reference to DAPort.

14.7.4 CIM_HostedAccessPoint

Associates ComputerSystem to SCSIProtocolEndpoint. Limit to targets. The class definition specializes the CIM_HostedAccessPoint definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 102 describes class CIM_HostedAccessPoint.

Table 102 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint

Properties	Flags	Requirement	Description and Notes
Antecedent (overridden)		Mandatory	Reference to controller ComputerSystem.
Dependent (overridden)		Mandatory	Reference to SCSIProtocolEndpoint.

14.7.5 CIM_SCSIProtocolEndpoint

ProtocolEndpoint representing support for SCSI. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 103 describes class CIM_SCSIProtocolEndpoint.

Table 103 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	
ProtocolIFTType		Mandatory	Shall be 1 (Other).
OtherTypeDescription (overridden)		Mandatory	Shall be the string 'SCSI'.
Role (added)		Mandatory	Shall be 3 (Target).
ConnectionType (added)		Mandatory	Shall be 1 (Other).
OtherConnectionType (added)		Mandatory	

14.7.6 CIM_SystemDevice (Port)

Associates controller ComputerSystem to DAPort. The class definition specializes the CIM_SystemDevice definition in the Generic Target Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 104 describes class CIM_SystemDevice (Port).

Table 104 - SMI Referenced Properties/Methods for CIM_SystemDevice (Port)

Properties	Flags	Requirement	Description and Notes
GroupComponent (overridden)		Mandatory	Reference to controller ComputerSystem.
PartComponent (overridden)		Mandatory	Reference to DAPort.

EXPERIMENTAL

EXPERIMENTAL

15 Generic Initiator Ports Profile

15.1 Synopsis

Profile name: Generic Initiator Ports

Version: 1.4.0

Organization: SNIA

CIM schema version: 2.9.0 (later schema versions may be required for specializations)

Central Class: CIM_LogicalPort

Scoping Class: a CIM_System in a separate autonomous profile

The Generic Initiator Port Profile models the generic management interfaces of initiator ports in host adaptors or storage systems.

This abstract profile specification shall not be directly implemented; implementations shall be based on a profile specification that specializes the requirements of this profile.

15.2 Description

The Generic Initiator Port Profile models the generic behavior of initiator ports in host adaptors. It uses the same primary classes as the Generic Target Port Profile (see 7 Generic Target Ports Profile).

15.3 Implementation

15.3.1 General

The initiator port is modeled as a ProtocolEndpoint connected to a LogicalPort.

The LogicalDevice instances may represent local storage (embedded in the system containing the initiator ports) or remote storage. When it represents remote storage the Name and NameFormat properties are used as correlatable ids to reference the remote device. When the LogicalDevice represents local disk storage, it may be represented as an instance of StorageVolume (subclass of LogicalDevice) or part of an instance of the Disk Drive Lite Profile. A property on LogicalPort called UsageRestriction is available to indicate whether the controller is capable of providing a “front end” (target), a “back end” (initiator), or both interfaces.

Figure 15 depicts the generic model.

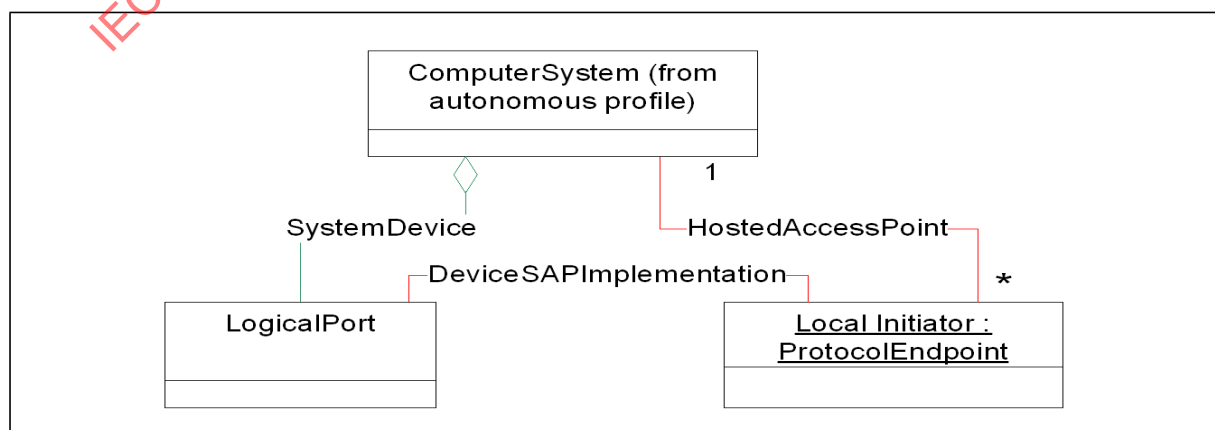


Figure 15 - Generic Initiator Port Model

15.3.2 Remote Device Models

15.3.2.1 General

The implementation may optionally include discovered remote elements. There are two optional approaches to modeling remote elements, depending on the capabilities of the underlying host drivers

The first approach is to model a collection of ports representing the local and remote ports that are known to be connected. This approach is appropriate for ATA device and when the underlying drivers or software is limited to information about remote ports and does not include details of the logical devices connected to remote ports. Figure 16 depicts the optional connectivity collection model.

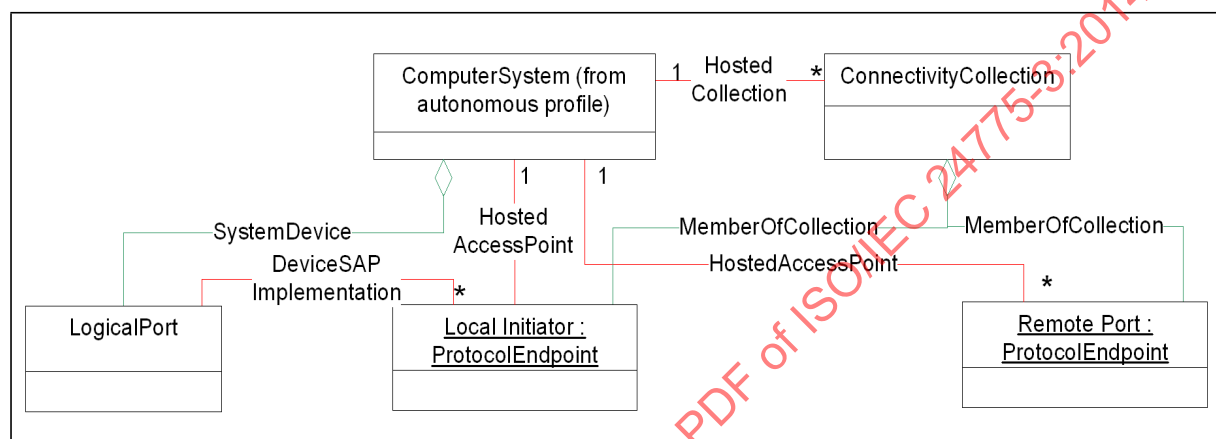


Figure 16 - Optional Connectivity Collection Model

The nature of membership in the collection varies with transports and configuration options. For example, in a parallel SCSI environment, the ConnectivityCollection includes all initiators/targets attached to the bus. In an FC fabric environment, the ConnectivityCollection contains ports that share a zone. In many cases, the ConnectivityCollection could include remote initiators as well as remote devices.

The second approach to modeling remote devices is to include the full initiator/target/logical-unit path model that describes multipath connectivity. This approach has the advantage of including the logical units and including the full path connectivity. The disadvantage is that some OSes handle multipath support in different components from HBA support, making it more efficient to provide the multipath model as part of the Host Discovered Resources Profile. Figure 17 depicts the optional full-path model.

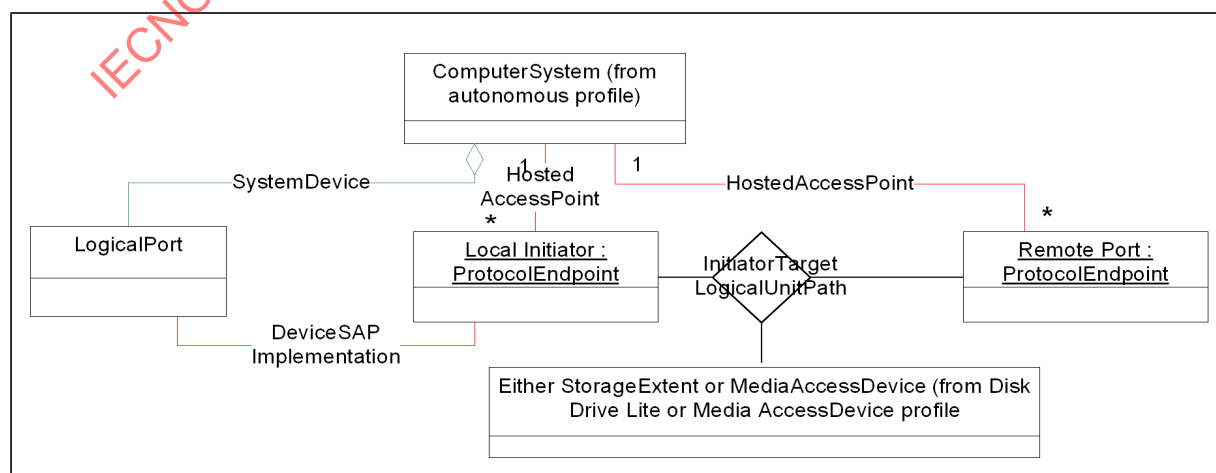


Figure 17 - Optional Full-Path Model

The instrumentation may support the full-path and connectivity collection options by making appropriate ProtocolEndpoints members of ConnectivityCollections.

15.3.2.2 Optional Model for Attached Disks

Disks are modeled using the full-path model and the Disk Drive Lite Profile. The appropriate subclass of InitiatorTargetLogicalUnitPath shall be dependent on whether the disks use the SCSI, ATA or SB commend set. This association references StorageExtent and initiator and target ProtocolEndpoints. The association also provides the disk's logical unit number. The target ProtocolEndpoint referenced from InitiatorTargetLogicalUnitPath shall be the ProtocolEndpoint from the Disk Drive Lite Profile associated indirectly to StorageExtent via DiskDrive. This is the same ProtocolEndpoint described as the optional remote ProtocolEndpoint in initiator ports profiles.

The ProtocolEndpoints may be associated to a ConnectivityCollection representing a collection of logically connected devices, as illustrated in Figure 18.

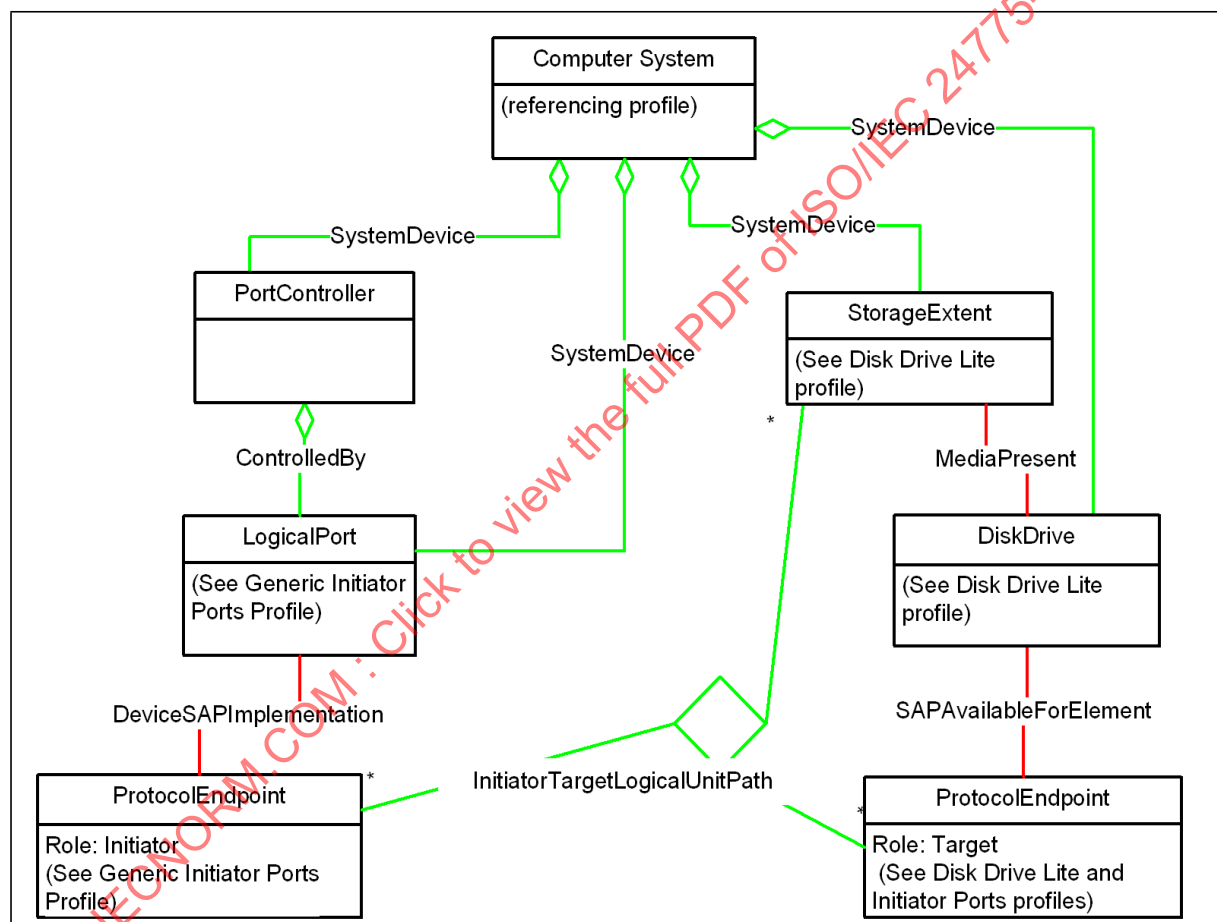


Figure 18 - HBA and Disk Model

15.3.2.3 Optional Model for attached Tape/CD/DVD Drives

The model, illustrated in Figure 19, and requirements are similar to those for disks (see 15.3.2.2), but use the Media Access Device Profile rather than Disk Drive Lite and the appropriate subclass of MediaAccessDevice rather than DiskDrive.

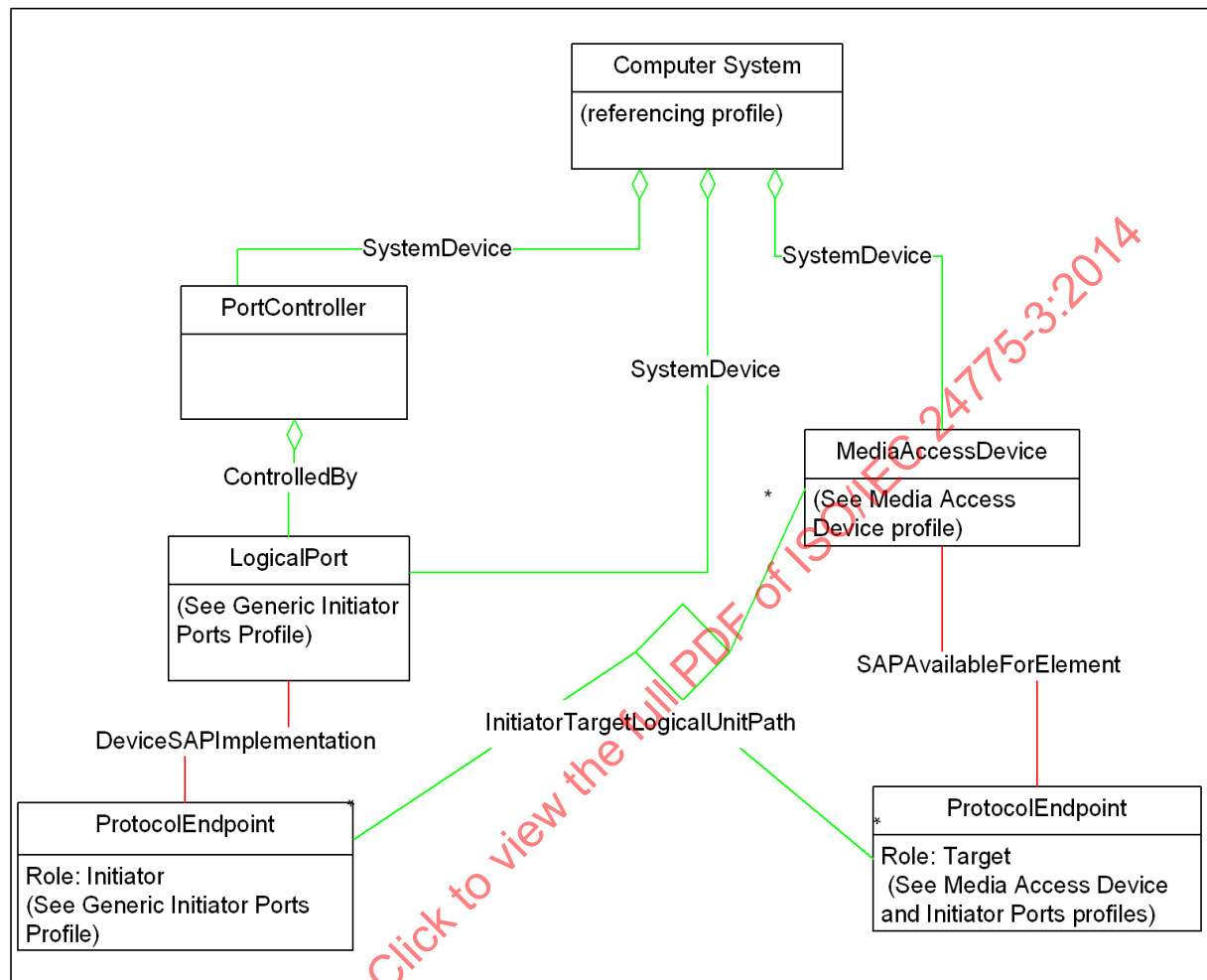


Figure 19 - HBA and Tape or Optical Devices

15.3.2.4 Optional Port Statistics

An implementation of an initiator port profile may optionally support port statistics. Figure 20 depicts the model for port statistics.

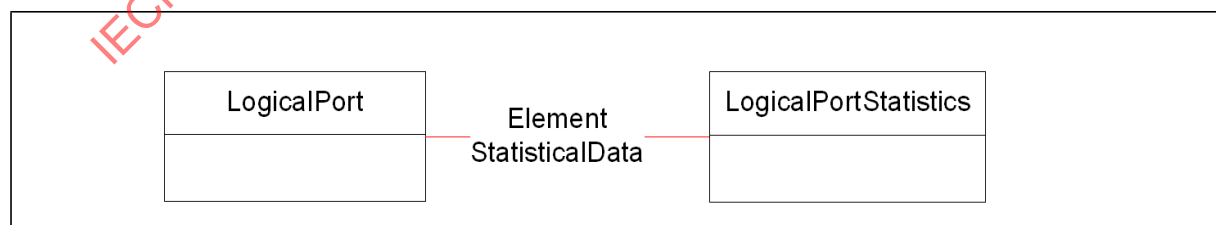


Figure 20 - Port Statistics

If a specialization of the profile specifies a subclass of LogicalPortStatistics (e.g., FCPortStatistics), the implementation should associate that subclass to the appropriate

subclass of LogicalPort (e.g., FCPort), as shown in Figure 21. Otherwise, an implementation should use LogicalPortStatistics directly.

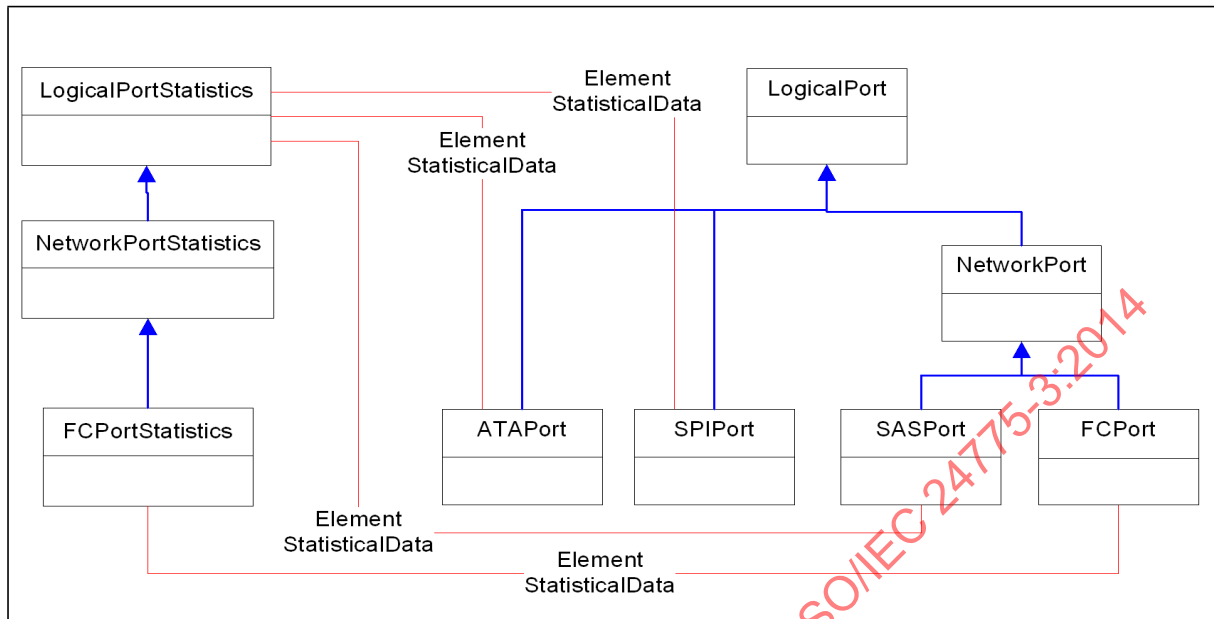


Figure 21 - Port Statistics Hierarchy

15.3.3 Health and Fault Management Considerations

Not defined in this International Standard.

15.3.4 Cascading Considerations

Not defined in this International Standard.

15.4 Methods

15.4.1 Extrinsic Methods of this Profile

None.

15.4.2 Intrinsic Methods of this Profile

The profile supports read methods and association traversal. Specifically, the list of intrinsic operations supported are as follows:

- GetInstance
- Associators
- AssociatorNames
- References
- ReferenceNames
- EnumerateInstances
- EnumerateInstanceNames

15.5 Use Cases

Not defined in this International Standard.

15.6 CIM Elements

15.6.1 General

Table 105 describes the CIM elements for Generic Initiator Ports.

Table 105 - CIM Elements for Generic Initiator Ports

Element Name	Requirement	Description
15.6.2 CIM_ConnectivityCollection	Optional	Represents a collection of connected ProtocolEndpoints.
15.6.3 CIM_DeviceSAPImplementation	Mandatory	Connects Initiator LogicalPort and ProtocolEndpoint.
15.6.4 CIM_ElementStatisticalData (Port Statistics)	Optional	Connects LogicalPort and LogicalPortStatistics.
15.6.5 CIM_HostedAccessPoint (Initiator)	Mandatory	Associates system to initiator protocol endpoints.
15.6.6 CIM_HostedAccessPoint (Target)	Optional	Associates system to optional remote protocol endpoints.
15.6.7 CIM_HostedCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Associates the ConnectivityCollection to the hosting System.
15.6.8 CIM_LogicalPort	Mandatory	Represents the logical aspects of the physical port and may have multiple associated protocols.
15.6.9 CIM_MemberOfCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Associates ProtocolEndpoints to the ConnectivityCollection.
15.6.10 CIM_ProtocolEndpoint (Initiator)	Mandatory	Represents a protocol (command set) supported by the port. The appropriate subclass (SCSIProtocolEndpoint, ATAProtocolEndpoint, SBProtocolEndpoint) should be used in initiator port specialized profiles.
15.6.11 CIM_ProtocolEndpoint (Target)	Optional	Models protocols of remote ports - target devices and possibly other initiators.
15.6.12 CIM_SystemDevice (Initiator Ports)	Mandatory	Associates system to initiator ports.
15.6.13 SNIA_LogicalPortStatistics	Optional	Statistics for a port.

15.6.2 CIM_ConnectivityCollection

Represents a collection of connected ProtocolEndpoints.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 106 describes class CIM_ConnectivityCollection.

Table 106 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection

Properties	Requirement	Description and Notes
InstanceID	Mandatory	

15.6.3 CIM_DeviceSAPImplementation

Connects Initiator LogicalPort and ProtocolEndpoint.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 107 describes class CIM_DeviceSAPImplementation.

Table 107 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Requirement	Description and Notes
Dependent	Mandatory	Reference to ProtocolEndpoint(Initiator).
Antecedent	Mandatory	Reference to LogicalPort.

15.6.4 CIM_ElementStatisticalData (Port Statistics)

Connects LogicalPort and LogicalPortStatistics.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 108 describes class CIM_ElementStatisticalData (Port Statistics).

Table 108 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)

Properties	Requirement	Description and Notes
ManagedElement	Mandatory	Reference to LogicalPort.
Stats	Mandatory	Reference to LogicalPortStatistics.

15.6.5 CIM_HostedAccessPoint (Initiator)

Associates system to initiator protocol endpoints.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 109 describes class CIM_HostedAccessPoint (Initiator).

Table 109 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent	Mandatory	Reference to ProtocolEndpoint(Initiator).

15.6.6 CIM_HostedAccessPoint (Target)

Associates system to optional remote protocol endpoints.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 110 describes class CIM_HostedAccessPoint (Target).

Table 110 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent	Mandatory	Reference to ProtocolEndpoint(Target).

15.6.7 CIM_HostedCollection (Connectivity Collection)

Associates the ConnectivityCollection to the hosting System.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 111 describes class CIM_HostedCollection (Connectivity Collection).

Table 111 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Dependent	Mandatory	Reference to ConnectivityCollection.
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.

15.6.8 CIM_LogicalPort

Represents the logical aspects of the physical port and may have multiple associated protocols.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 112 describes class CIM_LogicalPort.

Table 112 - SMI Referenced Properties/Methods for CIM_LogicalPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus	Mandatory	
UsageRestriction	Mandatory	Shall be 3 for ports restricted to back-end (initiator) only or 4 if the port is unrestricted.
PortType	Mandatory	Initiator port specialized profiles specify the appropriate subset of values.

15.6.9 CIM_MemberOfCollection (Connectivity Collection)

Associates ProtocolEndpoints to the ConnectivityCollection.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 113 describes class CIM_MemberOfCollection (Connectivity Collection).

Table 113 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Member	Mandatory	Reference to ProtocolEndpoint.
Collection	Mandatory	Reference to ConnectivityCollection.

15.6.10 CIM_ProtocolEndpoint (Initiator)

Represents a protocol (command set) supported by the port.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 114 describes class CIM_ProtocolEndpoint (Initiator).

Table 114 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint (Initiator)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	

Table 114 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint (Initiator)

Properties	Requirement	Description and Notes
Name	Mandatory	See <i>Information Technology – Storage Management, Part 2 Common Architecture</i> 8.6.3 Standard Formats for Port Names.
ProtocolIFType	Mandatory	Shall be 1 (Other).
OtherTypeDescription	Mandatory	Shall be the string 'SCSI', 'ATA', or 'SB'. Initiator port specialized profiles specify the appropriate subset.

15.6.11 CIM_ProtocolEndpoint (Target)

Models protocols of remote ports - target devices and possibly other initiators. The appropriate subclass (SCSIProtocolEndpoint, ATAProtocolEndpoint, SBProtocolEndpoint) should be used in initiator port specialized profiles.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 115 describes class CIM_ProtocolEndpoint (Target).

Table 115 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint (Target)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType	Mandatory	The values in MOFs map to IETF values and exclude storage. Shall be 1 (Other) and set OtherTypeDescription appropriately.
OtherTypeDescription	Mandatory	Shall be the string 'SCSI', 'ATA', or 'SB'. Initiator port specialized profiles specify the appropriate subset.

15.6.12 CIM_SystemDevice (Initiator Ports)

Associates system to initiator ports.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 116 describes class CIM_SystemDevice (Initiator Ports).

Table 116 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to ComputerSystem.
PartComponent	Mandatory	Reference to LogicalPort.

15.6.13 SNIA_LogicalPortStatistics

Statistics for a port.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 117 describes class SNIA_LogicalPortStatistics.

Table 117 - SMI Referenced Properties/Methods for SNIA_LogicalPortStatistics

Properties	Requirement	Description and Notes
ElementName	Mandatory	
InstanceID	Mandatory	
BytesTransmitted	Mandatory	
BytesReceived	Mandatory	
PacketsTransmitted	Mandatory	
PacketsReceived	Mandatory	

EXPERIMENTAL

EXPERIMENTAL

16 Parallel SCSI (SPI) Initiator Ports Profile

16.1 Synopsis

Profile Name: SPI Initiator Ports (Component Profile)

Version: 1.4.0

Organization: SNIA

CIM Schema Version: 2.11.0

Related Profiles for SPI Initiator Ports: Not defined in this International Standard.

Specializes: Generic Initiator Port Profile

Central Class: CIM_SPIPort

Scoping Class: a CIM_System in a separate autonomous profile

The SPI Initiator Ports Profile models the behavior of a parallel SCSI (SPI) initiator port.

16.2 Description

The SPI Initiator Port Profile defines the model to parallel SCSI ports.

16.3 Implementation

16.3.1 General

A typical instance diagram is provided in Figure 22.

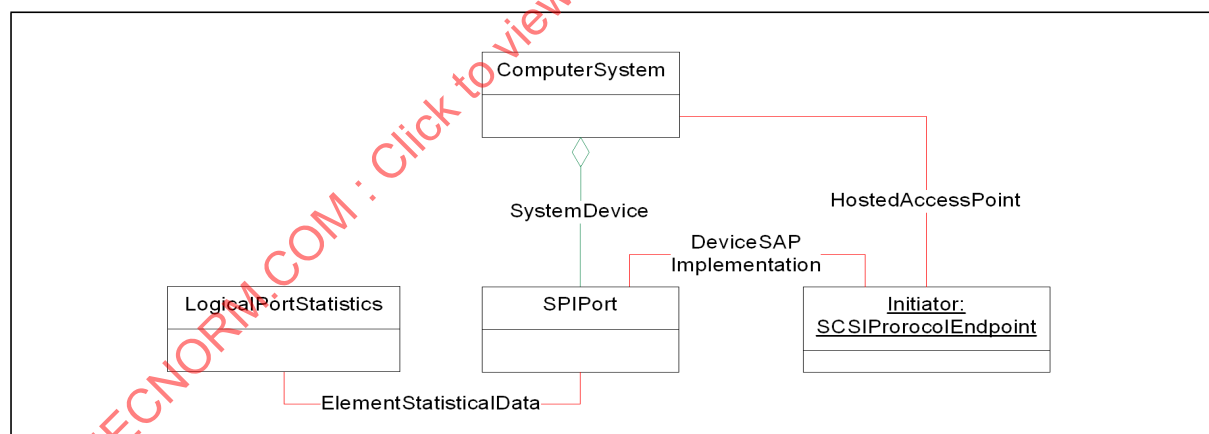


Figure 22 - SPI Initiator Port Instance Diagram

16.3.2 Health and Fault Management Considerations

Table 118 summarizes the Health and Fault Management issues that are unique to this profile.

Table 118 - SPIPort OperationalStatus

OperationalStatus	Description
OK	Port is online
Error	Port has a failure

Table 118 - SPIPort OperationalStatus

OperationalStatus	Description
Stopped	Port is disabled
InService	Port is in Self Test
Unknown	

16.3.3 Cascading Considerations

Not defined in this International Standard.

16.4 Methods

16.4.1 Extrinsic Methods of this Profile

None.

16.4.2 Intrinsic Methods of this Profile

The profile supports read methods and association traversal. Specifically, the list of intrinsic operations supported are as follows:

- GetInstance
- Associators
- AssociatorNames
- References
- ReferenceNames
- EnumerateInstances
- EnumerateInstanceNames

16.5 Detailed Use Cases and Recipes

None.

16.6 CIM Elements

16.6.1 General

Table 119 describes the CIM elements for SPI Initiator Ports.

Table 119 - CIM Elements for SPI Initiator Ports

Element Name	Requirement	Description
16.6.2 CIM_ConnectivityCollection	Mandatory	Represents a collection of connected SCSIProtocolEndpoints.
16.6.3 CIM_DeviceSAPImplementation	Mandatory	Connects Initiator SPILogicalPort and SCSIProtocolEndpoint.
16.6.4 CIM_ElementStatisticalData (Port Statistics)	Mandatory	Connects SPIPort and LogicalPortStatistics.
16.6.5 CIM_HostedAccessPoint (Initiator)	Mandatory	Associates system to initiator protocol endpoints.
16.6.6 CIM_HostedAccessPoint (Target)	Optional	Associates system to optional remote protocol endpoints.

Table 119 - CIM Elements for SPI Initiator Ports

Element Name	Requirement	Description
16.6.7 CIM_HostedCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Associates the ConnectivityCollection to the hosting System.
16.6.8 CIM_MemberOfCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Represents a collection of connected SCSIProtocolEndpoints.
16.6.9 CIM_SCSIInitiatorTargetLogicalUnitPath	Optional	Represents a path between a SCSI initiator, target, and logical unit.
16.6.10 CIM_SCSIProtocolEndpoint (Initiator)	Mandatory	Represents support for the SCSI command set.
16.6.11 CIM_SCSIProtocolEndpoint (Target)	Mandatory	Models protocols of remote ports - target devices and possibly other initiators.
16.6.12 CIM_SPIPort	Mandatory	Represents the logical aspects of the physical port and may have multiple associated protocols.
16.6.13 CIM_SystemDevice (Initiator Ports)	Mandatory	Associates system to initiator ports.
16.6.14 SNIA_LogicalPortStatistics	Optional	Statistics for a port.

16.6.2 CIM_ConnectivityCollection

Represents a collection of connected SCSIProtocolEndpoints. The class definition specializes the CIM_ConnectivityCollection definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 120 describes class CIM_ConnectivityCollection.

Table 120 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection

Properties	Requirement	Description and Notes
InstanceID	Mandatory	

16.6.3 CIM_DeviceSAPImplementation

Connects Initiator SPILogicalPort and SCSIProtocolEndpoint. The class definition specializes the CIM_DeviceSAPImplementation definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 121 describes class CIM_DeviceSAPImplementation.

Table 121 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Requirement	Description and Notes
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint(Initiator).
Antecedent (overridden)	Mandatory	Reference to SPIPort.

16.6.4 CIM_ElementStatisticalData (Port Statistics)

Connects SPIPort and LogicalPortStatistics. The class definition specializes the CIM_ElementStatisticalData definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 122 describes class CIM_ElementStatisticalData (Port Statistics).

Table 122 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)

Properties	Requirement	Description and Notes
ManagedElement (overridden)	Mandatory	Reference to SPIPort.
Stats	Mandatory	Reference to LogicalPortStatistics.

16.6.5 CIM_HostedAccessPoint (Initiator)

Associates system to initiator protocol endpoints. The class definition specializes the CIM_HostedAccessPoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 123 describes class CIM_HostedAccessPoint (Initiator).

Table 123 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint(Initiator).

16.6.6 CIM_HostedAccessPoint (Target)

Associates system to optional remote protocol endpoints. The class definition specializes the CIM_HostedAccessPoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 124 describes class CIM_HostedAccessPoint (Target).

Table 124 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint(Target).

16.6.7 CIM_HostedCollection (Connectivity Collection)

Associates the ConnectivityCollection to the hosting System.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 125 describes class CIM_HostedCollection (Connectivity Collection).

Table 125 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Dependent	Mandatory	Reference to ConnectivityCollection.
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.

16.6.8 CIM_MemberOfCollection (Connectivity Collection)

Represents a collection of connected SCSIProtocolEndpoints. The class definition specializes the CIM_MemberOfCollection definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 126 describes class CIM_MemberOfCollection (Connectivity Collection).

Table 126 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Member (overridden)	Mandatory	Reference to SCSIProtocolEndpoint(Initiator or Target).
Collection	Mandatory	Reference to ConnectivityCollection.

16.6.9 CIM_SCSIInitiatorTargetLogicalUnitPath

Represents a path between a SCSI initiator, target, and logical unit.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 127 describes class CIM_SCSIInitiatorTargetLogicalUnitPath.

Table 127 - SMI Referenced Properties/Methods for CIM_SCSIInitiatorTargetLogicalUnitPath

Properties	Requirement	Description and Notes
LogicalUnit	Mandatory	Reference to StorageExtent in Disk Drive Lite Profile or MediaAccessDevice in Media Access Device Profile.
Target	Mandatory	Reference to SCSIProtocolEndpoint(Target).
Initiator	Mandatory	Reference to SCSIProtocolEndpoint(Initiator).

16.6.10 CIM_SCSIProtocolEndpoint (Initiator)

Represents support for the SCSI command set. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 128 describes class CIM_SCSIProtocolEndpoint (Initiator).

Table 128 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Initiator)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	See <i>Information Technology – Storage Management, Part 2 Common Architecture</i> 8.6.3 Standard Formats for Port Names.

Table 128 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Initiator)

Properties	Requirement	Description and Notes
ProtocolIFType	Mandatory	Shall be 1 (Other).
OtherTypeDescription	Mandatory	Shall be the string 'SCSI', 'ATA', or 'SB'. Initiator port specialized profiles specify the appropriate subset.
ConnectionType (added)	Mandatory	Shall be 3 (Parallel SCSI).
Role (added)	Mandatory	Shall be 2 (Initiator).

16.6.11 CIM_SCSIProtocolEndpoint (Target)

Models protocols of remote ports - target devices and possibly other initiators. The appropriate subclass (SCSIProtocolEndpoint, ATAProtocolEndpoint, SBProtocolEndpoint) should be used in initiator port specialized profiles. The class definition specializes the CIM_LogicalPort definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 129 describes class CIM_SCSIProtocolEndpoint (Target).

Table 129 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Target)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType	Mandatory	The values in MOFs map to IETF values and exclude storage. Shall be 1 (Other) and set OtherTypeDescription appropriately.
OtherTypeDescription	Mandatory	Shall be the string 'SCSI', 'ATA', or 'SB'. Initiator port specialized profiles specify the appropriate subset.
ConnectionType (added)	Mandatory	Shall be 3 (Parallel SCSI).

16.6.12 CIM_SPIPort

Represents the logical aspects of the physical port and may have multiple associated protocols. The class definition specializes the CIM_LogicalPort definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 130 describes class CIM_SPIPort.

Table 130 - SMI Referenced Properties/Methods for CIM_SPIPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus (overridden)	Mandatory	Shall be 0 (Unknown), 2 (OK), 6 (Error), 10 (Stopped), or 11 (In Service).
UsageRestriction (overridden)	Mandatory	Shall be 3 (Back-end Only).
PortType (overridden)	Mandatory	Shall be 140 (SCSI Parallel).

16.6.13 CIM_SystemDevice (Initiator Ports)

Associates system to initiator ports. The class definition specializes the CIM_SystemDevice definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 131 describes class CIM_SystemDevice (Initiator Ports).

Table 131 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to ComputerSystem.
PartComponent (overridden)	Mandatory	Reference to SPIPort(Initiator).

16.6.14 SNIA_LogicalPortStatistics

Statistics for a port.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 132 describes class SNIA_LogicalPortStatistics.

Table 132 - SMI Referenced Properties/Methods for SNIA_LogicalPortStatistics

Properties	Requirement	Description and Notes
ElementName	Mandatory	
InstanceID	Mandatory	
BytesTransmitted	Mandatory	
BytesReceived	Mandatory	
PacketsTransmitted	Mandatory	
PacketsReceived	Mandatory	

EXPERIMENTAL

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

EXPERIMENTAL

17 iSCSI Initiator Port Profile

17.1 Synopsis

Profile Name: iSCSI Initiator Ports (Component Profile)

Version: 1.2.0

Organization: SNIA

CIM Schema Version: 2.13.1

Table 133 describes the related profiles for iSCSI Initiator Ports.

Table 133 - Related Profiles for iSCSI Initiator Ports

Profile Name	Organization	Version	Requirement	Description
Indication	SNIA	1.5.0	Mandatory	

Specializes: Generic Initiator Port Profile

Central Class: CIM_EthernetPort

Scoping Class: a CIM_System in a separate autonomous profile

Models an adapter (NIC, HBA, TOE) for iSCSI.

17.2 Description

Models an adapter (NIC, HBA, TOE) for iSCSI.

17.3 Implementation

17.3.1 General

Other port profiles have a single physical port (LogicalPort subclass) associated with each SCSI initiator (SCSIProtocolEndpoint). iSCSI allows multiple connections (each with a single Ethernet port) in a session that acts as a SCSI initiator. This profile includes the subset of classes that model the SCSI initiator and its relationship to logical classes that model physical elements (Ethernet ports).

Figure 23 depicts a configuration with an initiator with two Ethernet ports that are part of a single session that acts as a SCSI initiator. The Ethernet ports (referred to in iSCSI literature as Network Portals) are modeled as instances of EthernetPort, IPProtocolEndpoint, and TCPProtocolEndpoint with 1-1 cardinality. These ports are in the initiator side, the target ports are not required in this profile. Note that all ProtocolEndpoint instances need a HostAccessPoint association to the ComputerSystem, some are omitted to keep the diagram less cluttered.

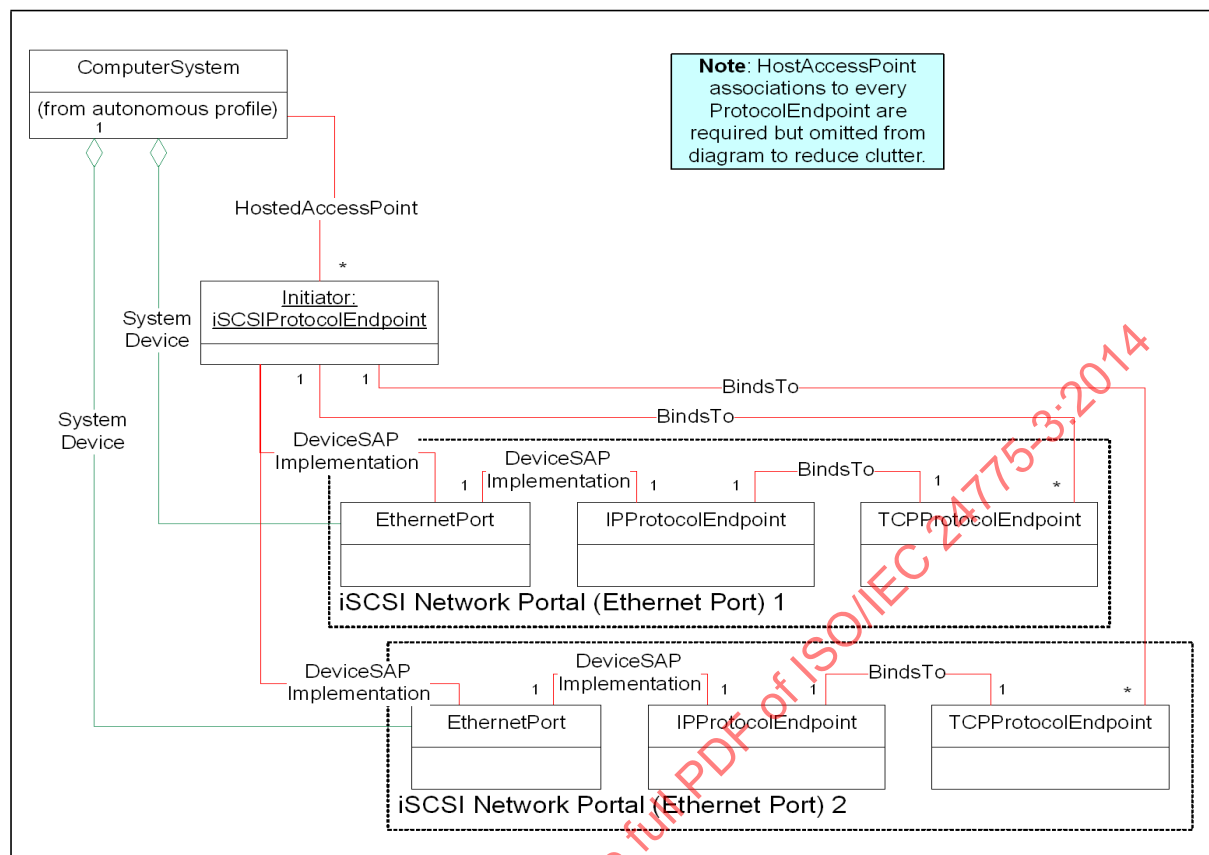


Figure 23 - iSCSI Initiator Port Instance Diagram

17.3.2 Health and Fault Management Considerations

Table 134 describes EthernetPort OperationalStatus.

Table 134 - EthernetPort OperationalStatus

OperationalStatus	Description
OK	Port is online
Error	Port has a failure
Stopped	Port is disabled
InService	Port is in Self Test
Unknown	

17.3.3 Cascading Considerations

Not defined in this International Standard.

17.4 Methods

17.4.1 Extrinsic Methods of this Profile

None

17.4.2 Intrinsic Methods of this Profile

The profile supports read methods and association traversal. Specifically, the list of intrinsic operations supported are as follows:

- GetInstance
- Associators
- AssociatorNames
- References
- ReferenceNames
- EnumerateInstances
- EnumerateInstanceNames

17.5 Detailed Use Cases and Recipes

None.

17.6 CIM Elements

17.6.1 General

Table 135 describes the CIM elements for iSCSI Initiator Ports.

Table 135 - CIM Elements for iSCSI Initiator Ports

Element Name	Requirement	Description
17.6.2 CIM_BindsTo (Host Hardware RAID Controller)	Mandatory	
17.6.3 CIM_DeviceSAPImplementation (IPProtocolEndpoint to EthernetPort)	Mandatory	
17.6.4 CIM_DeviceSAPImplementation (iSSIPProtocolEndpoint to EthernetPort)	Mandatory	
17.6.5 CIM_EthernetPort (Host Hardware RAID Controller)	Mandatory	
17.6.6 CIM_HostedAccessPoint (System to IPProtocolEndpoint)	Mandatory	
17.6.7 CIM_HostedAccessPoint (System to TCPProtocolEndpoint)	Mandatory	
17.6.8 CIM_HostedAccessPoint (System to iSSIPProtocolEndpoint)	Mandatory	
17.6.9 CIM_IPProtocolEndpoint (Host Hardware RAID Controller)	Mandatory	
17.6.10 CIM_LogicalDevice (Host Hardware RAID Controller)	Optional	
17.6.11 CIM_SystemDevice (System to EthernetPort)	Mandatory	
17.6.12 CIM_SystemDevice (System to LogicalDevice)	Mandatory	
17.6.13 CIM_TCPProtocolEndpoint (Host Hardware RAID Controller)	Mandatory	
17.6.14 CIM_iSSIPProtocolEndpoint (Host Hardware RAID Controller)	Mandatory	

Table 135 - CIM Elements for iSCSI Initiator Ports

Element Name	Requirement	Description
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_EthernetPort	Mandatory	Port Creation.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_EthernetPort AND SourceInstance.CIM_EthernetPort::OperationalStatus <> PreviousInstance.CIM_EthernetPort::OperationalStatus	Mandatory	CQL -Port Status Change.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_EthernetPort	Mandatory	Port Removal.

17.6.2 CIM_BindsTo (Host Hardware RAID Controller)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 136 describes class CIM_BindsTo (Host Hardware RAID Controller).

**Table 136 - SMI Referenced Properties/Methods for CIM_BindsTo
(Host Hardware RAID Controller)**

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

17.6.3 CIM_DeviceSAPImplementation (IPProtocolEndpoint to EthernetPort)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 137 describes class CIM_DeviceSAPImplementation (IPProtocolEndpoint to EthernetPort).

**Table 137 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation
(IPProtocolEndpoint to EthernetPort)**

Properties	Requirement	Description and Notes
Dependent	Mandatory	
Antecedent	Mandatory	

17.6.4 CIM_DeviceSAPImplementation (iSSIPProtocolEndpoint to EthernetPort)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 138 describes class CIM_DeviceSAPImplementation (iSSIPProtocolEndpoint to EthernetPort).

Table 138 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation (iSSIPProtocolEndpoint to EthernetPort)

Properties	Requirement	Description and Notes
Dependent	Mandatory	
Antecedent	Mandatory	

17.6.5 CIM_EthernetPort (Host Hardware RAID Controller)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 139 describes class CIM_EthernetPort (Host Hardware RAID Controller).

Table 139 - SMI Referenced Properties/Methods for CIM_EthernetPort (Host Hardware RAID Controller)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
PortType	Mandatory	
OperationalStatus	Mandatory	
PermanentAddress	Mandatory	

17.6.6 CIM_HostedAccessPoint (System to IPProtocolEndpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 140 describes class CIM_HostedAccessPoint (System to IPProtocolEndpoint).

Table 140 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (System to IPProtocolEndpoint)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

17.6.7 CIM_HostedAccessPoint (System to TCPProtocolEndpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 141 describes class CIM_HostedAccessPoint (System to TCPProtocolEndpoint).

Table 141 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (System to TCPProtocolEndpoint)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

17.6.8 CIM_HostedAccessPoint (System to iSCSIProtocolEndpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 142 describes class CIM_HostedAccessPoint (System to iSCSIProtocolEndpoint).

Table 142 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (System to iSCSIProtocolEndpoint)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

17.6.9 CIM_IPProtocolEndpoint (Host Hardware RAID Controller)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 143 describes class CIM_IPProtocolEndpoint (Host Hardware RAID Controller).

Table 143 - SMI Referenced Properties/Methods for CIM_IPProtocolEndpoint (Host Hardware RAID Controller)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
IPv4Address	Optional	Maps to IMA_NETWORK_PORTAL_PROPERTIES, ipAddress.
IPv6Address	Optional	Maps to IMA_NETWORK_PORTAL_PROPERTIES, ipAddress.
ProtocolIFType	Mandatory	

17.6.10 CIM_LogicalDevice (Host Hardware RAID Controller)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 144 describes class CIM_LogicalDevice (Host Hardware RAID Controller).

**Table 144 - SMI Referenced Properties/Methods for CIM_LogicalDevice
(Host Hardware RAID Controller)**

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
Name	Mandatory	
OperationalStatus	Mandatory	

17.6.11 CIM_SystemDevice (System to EthernetPort)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 145 describes class CIM_SystemDevice (System to EthernetPort).

**Table 145 - SMI Referenced Properties/Methods for CIM_SystemDevice
(System to EthernetPort)**

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	
PartComponent	Mandatory	

17.6.12 CIM_SystemDevice (System to LogicalDevice)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 146 describes class CIM_SystemDevice (System to LogicalDevice).

Table 146 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to LogicalDevice)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	
PartComponent	Mandatory	

17.6.13 CIM_TCPProtocolEndpoint (Host Hardware RAID Controller)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 147 describes class CIM_TCPProtocolEndpoint (Host Hardware RAID Controller).

Table 147 - SMI Referenced Properties/Methods for CIM_TCPProtocolEndpoint (Host Hardware RAID Controller)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
PortNumber	Mandatory	
ProtocolIFType	Mandatory	

17.6.14 CIM_iSCSIProtocolEndpoint (Host Hardware RAID Controller)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 148 describes class CIM_iSCSIProtocolEndpoint (Host Hardware RAID Controller).

Table 148 - SMI Referenced Properties/Methods for CIM_iSCSIProtocolEndpoint (Host Hardware RAID Controller)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType	Mandatory	Other.
OtherTypeDescription	Mandatory	
ConnectionType	Mandatory	iSCSI.
Role	Mandatory	Shall be 2 (Initiator).
Identifier	Mandatory	ISID.

EXPERIMENTAL

STABLE**18 FC Initiator Ports Profile****18.1 Synopsis****Profile Name:** FC Initiator Ports (Component Profile)**Version:** 1.4.0**Organization:** SNIA**CIM Schema Version:** 2.9.0

Table 149 describes the related profiles for FC Initiator Ports.

Table 149 - Related Profiles for FC Initiator Ports

Profile Name	Organization	Version	Requirement	Description
Indication	SNIA	1.5.0	Mandatory	

Specializes: Generic Initiator Ports Profile**Central Class:** CIM_FCPort**Scoping Class:** a CIM_System in a referencing autonomous profile

The FC Initiator Ports Profile models the behavior of a Fibre Channel port supporting FCP (SCSI command protocol).

18.2 Description

The FC Initiator Ports Profile models the behavior of a Fibre Channel port supporting FCP (SCSI command protocol).

18.3 Implementation**18.3.1 General**

Figure 24 is an example of a single port and drive connected to a single system using Fibre Channel. This instance diagram shows a disk (LogicalDevice in the diagram would be subclassed as something like StorageExtent) in an array, connected by a Fibre Channel port. The full model for the disk is shown in *Information Technology – Storage Management, Part 4 Block Devices* 12 Disk Drive Lite Subprofile. SCSIProtocolController is not generally used in initiator contexts. It is included here to be compatible with SMI-S 1.0 clients.

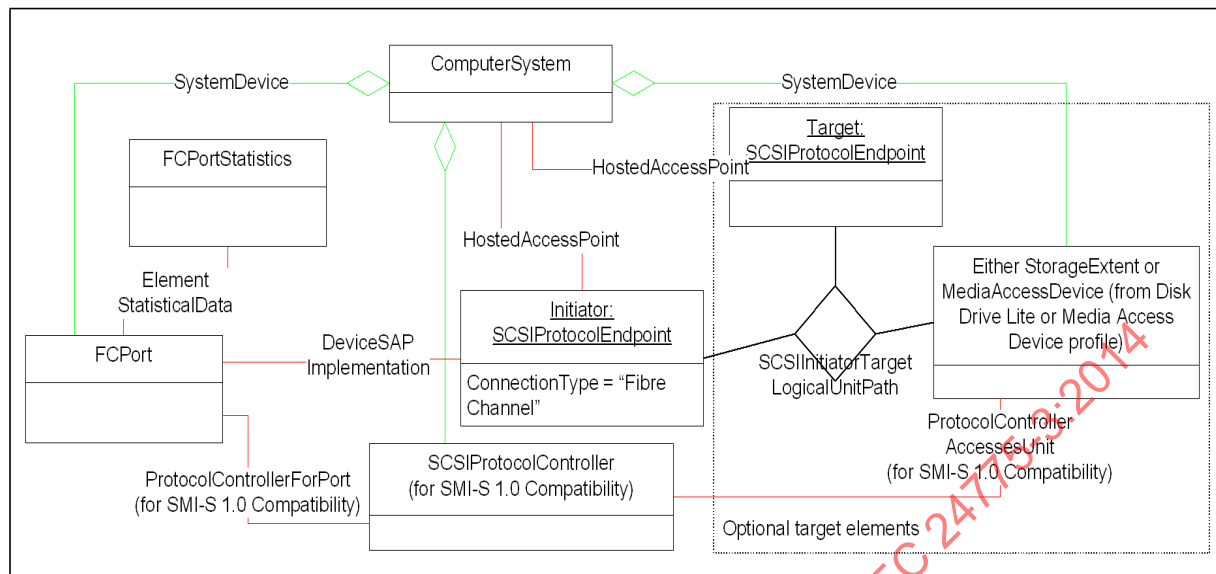


Figure 24 - Fibre Channel Initiator Instance Diagram

18.3.2 Port Statistics

The FCPortStatistics subclass of NetworkPortStatistics is optional. If supported, FCPortStatistics shall be associated to FcPort using ElementStatisticalData.

18.3.3 Logical Port Group (FC Node)

LogicalPortGroup may optionally be used to model the collection of ports that shared a Node WWN (in this case, both ports on a card, but other implementations are in use). If LogicalPortGroup is instantiated, it shall be associated to the ComputerSystem in the referencing profile using HostedCollection and also associated to FCPorts using MemberOfCollection. Figure 25, "FC Node Model" shows to model for FC Nodes.

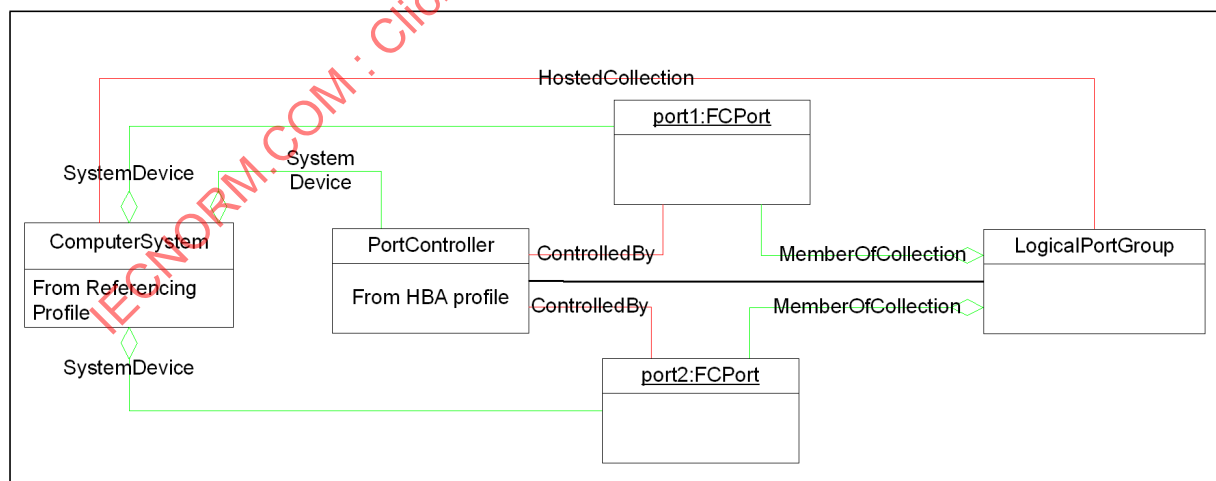


Figure 25 - FC Node Model

18.3.4 Health and Fault Management Considerations

Table 150 summarized the Health and Fault Management considerations specific to this profile.

Table 150 - FCPort OperationalStatus

OperationalStatus	Description
OK	Port is online
Error	Port has a failure
Stopped	Port is disabled
InService	Port is in Self Test
Unknown	

18.3.5 Cascading Considerations

Not defined in this International Standard.

18.4 Methods

18.4.1 Extrinsic Methods of this Profile

None.

18.4.2 Intrinsic Methods of this Profile

The profile supports read methods and association traversal. Specifically, the list of intrinsic operations supported are as follows:

- GetInstance
- Associators
- AssociatorNames
- References
- ReferenceNames
- EnumerateInstances
- EnumerateInstanceNames

18.5 Use Cases – Get the statistics for each FC port

This recipe is optional and assumes an FCPortStatistics instance is defined for each FCPort instance.

```
//
// DESCRIPTION
//
// Find the FCPortStatistics associated with FC ports
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
//
// 1. A reference to the top-level ComputerSystem in the FC HBA Profile,
//    which represents the system hosting the HBA, is known as $Host->
//
// Get a list of all the ports
$Ports->[] = AssociatorNames($Host->, // ObjectName
```

```

"CIM_SystemDevice", // AssocClass
"CIM_FCPort", // ResultClass
"GroupComponent", // Role
"PartComponent") // ResultRole

if ($Ports->[] == null || $Ports->[].length == 0) {
    <ERROR! No FC Ports on the host system!>
}

for (#i in $Ports->[] ) {
    // Get a list of FCPortStatistics associated with each port
    // Should only be exactly one FCPortStatistics instance
    $Stats->[] = AssociatorNames($Ports->[#i], // ObjectName
        "CIM_ElementStatisticalData", // AssocClass
        "CIM_FCPortStatistics", // ResultClass
        "ManagedElement", // Role
        "Stats") // ResultRole
    if ($Stats->[] == null || $Stats->[].length == 0) {
        <ERROR! Each FCPort shall have an associated FCPortStatistics>
    } else {
        if ( $Stats->[].length > 1) {
            <ERROR: More than 1 FCPortStatistics associated with a port>
        }
    }
    // $Stats[0]-> holds that stats
}

```

18.6 CIM Elements

18.6.1 General

Table 151 describes the CIM elements for FC Initiator Ports.

Table 151 - CIM Elements for FC Initiator Ports

Element Name	Requirement	Description
18.6.2 CIM_ConnectivityCollection	Optional	Represents a collection of connected ProtocolEndpoints.
18.6.3 CIM_DeviceSAPImplementation	Mandatory	Connects Initiator FCPort and SCSIProtocolEndpoint.
18.6.4 CIM_ElementStatisticalData (Port Statistics)	Conditional	Conditional requirement: support for the FC HBA profile. Connects FCPort and FCPortStatistics.
18.6.5 CIM_FCPort	Mandatory	Represents the logical aspects of the physical port and may have multiple associated protocols.
18.6.6 CIM_FCPortStatistics	Conditional	Conditional requirement: support for the FC HBA profile. Statistics for a port.
18.6.7 CIM_HostedAccessPoint (Initiator)	Mandatory	Associates system to initiator protocol endpoints.
18.6.8 CIM_HostedAccessPoint (Target)	Optional	Associates system to optional remote protocol endpoints.

Table 151 - CIM Elements for FC Initiator Ports

Element Name	Requirement	Description
18.6.9 CIM_HostedCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Associates the ConnectivityCollection to the hosting System.
18.6.10 CIM_MemberOfCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Associates SCSIProtocolEndpoints to the ConnectivityCollection.
18.6.11 CIM_ProtocolControllerForPort	Optional	Deprecated. Associates SCSIProtocolController to FCPort.
18.6.12 CIM_SCSIInitiatorTargetLogicalUnitPath	Optional	Represents a path between a SCSI initiator, target, and logical unit.
18.6.13 CIM_SCSIProtocolController	Optional	Deprecated. Represents a SCSI logical unit inventory.
18.6.14 CIM_SCSIProtocolEndpoint (Initiator)	Mandatory	Represents support for the SCSI command set.
18.6.15 CIM_SCSIProtocolEndpoint (Target)	Optional	Models remote ports - target devices and possibly other initiators.
18.6.16 CIM_SystemDevice (Initiator Ports)	Mandatory	Associates system to initiator ports.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_FCPort	Optional	Create FCPort.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.OperationalStatus <>PreviousInstance.OperationalStatus	Optional	Deprecated WQL -Modify FCPort.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.CIM_FCPort::OperationalStatus <>PreviousInstance.CIM_FCPort::OperationalStatus	Optional	CQL -Modify FCPort.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_FCPort	Optional	Delete FCPort.

18.6.2 CIM_ConnectivityCollection

Represents a collection of connected ProtocolEndpoints.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 152 describes class CIM_ConnectivityCollection.

Table 152 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection

Properties	Requirement	Description and Notes
InstanceID	Mandatory	

18.6.3 CIM_DeviceSAPImplementation

Connects Initiator FCPort and SCSIProtocolEndpoint. The class definition specializes the CIM_DeviceSAPImplementation definition in the Generic Initiator Ports profile. Properties or

methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 153 describes class CIM_DeviceSAPImplementation.

Table 153 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Requirement	Description and Notes
Dependent (overridden)	Mandatory	Reference to Initiator SCSIProtocolEndpoint.
Antecedent (overridden)	Mandatory	Reference to FCPort.

18.6.4 CIM_ElementStatisticalData (Port Statistics)

Connects FCPort and FCPortStatistics. The class definition specializes the CIM_ElementStatisticalData definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: support for the FC HBA profile.

Table 154 describes class CIM_ElementStatisticalData (Port Statistics).

Table 154 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)

Properties	Requirement	Description and Notes
ManagedElement (overridden)	Mandatory	Reference to FCPort.
Stats (overridden)	Mandatory	Reference to FCPortStatistics.

18.6.5 CIM_FCPort

Represents the logical aspects of the physical port and may have multiple associated protocols. The class definition specializes the CIM_LogicalPort definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 155 describes class CIM_FCPort.

Table 155 - SMI Referenced Properties/Methods for CIM_FCPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus (overridden)	Mandatory	Shall be 0 (Unknown), 2 (OK), 6 (Error), 10 (Stopped), or 11 (In Service).
UsageRestriction	Mandatory	Shall be 3 for ports restricted to back-end (initiator) only or 4 if the port is unrestricted.
PortType (overridden)	Mandatory	Shall be 0 1 10 11 12 13 14 15 16 17 18 (Unknown or Other or N or NL or F/NL or Nx or E or F or FL or B or G).
ElementName (added)	Mandatory	Port Symbolic Name.
Speed (added)	Mandatory	Speed in bits per second. Shall be 0, 1062500000 (1GFC), 2125000000 (2GFC), 4250000000 (4GFC), 8500000000 (8GFC), 10518750000 (10GFC), 14025000000 (16GFC), 21037500000 (20GFC) or 28500000000 (32GFC).
MaxSpeed (added)	Mandatory	Maximum Port Speed.
PortNumber (added)	Optional	
PermanentAddress (added)	Optional	Port WWN. PermanentAddress is optional when used as a back-end port in a device. This may be overridden in profiles that use this profile. Shall be 16 un-separated upper case hex digits. See <i>Information Technology – Storage Management, Part 2 Common Architecture</i> 8.6.3 Standard Formats for Port Names.
NetworkAddresses (added)	Optional	For Fibre Channel end device ports, the Fibre Channel ID. Shall be 16 un-separated upper case hex digits.
SupportedCOS (added)	Optional	Shall be 0 (unknown), 1 (Class 1), 2 (Class 2), 3, (Class 3), 4 (Class 4), 6 (Class 6), or 7 (Class 7).
ActiveCOS (added)	Optional	Shall be 0 (unknown), 1 (Class 1), 2 (Class 2), 3, (Class 3), 4 (Class 4), 6 (Class 6), or 7 (Class 7).
SupportedFC4Types (added)	Optional	
ActiveFC4Types (added)	Optional	
LinkTechnology (added)	Mandatory	Shall be 4 (FC).
SupportedMaximumTransmissionUnit (added)	Mandatory	
ActiveMaximumTransmissionUnit (added)	Optional	

18.6.6 CIM_FCPortStatistics

Statistics for a port. The class definition specializes the SNIA_LogicalPortStatistics definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: support for the FC HBA profile.

Table 156 describes class CIM_FCPortStatistics.

Table 156 - SMI Referenced Properties/Methods for CIM_FCPortStatistics

Properties	Requirement	Description and Notes
ElementName	Mandatory	
InstanceID	Mandatory	
BytesTransmitted (overridden)	Mandatory	From NetworkPortStatistics Superclass. Maps to HBA_PortStatistics.TxWords. Multiply word count by 4.
BytesReceived (overridden)	Mandatory	From NetworkPortStatistics Superclass. Maps to HBA_PortStatistics.RxWords. Multiply word count by 4.
PacketsTransmitted (overridden)	Mandatory	From NetworkPortStatistics Superclass. Maps to HBA_PortStatistics.TxFrames.
PacketsReceived (overridden)	Mandatory	From NetworkPortStatistics Superclass. Maps to HBA_PortStatistics.RxFrames.
CRCErrors (added)	Mandatory	Maps to HBA_PortStatistics.InvalidCRCCCount.
LinkFailures (added)	Mandatory	Maps to HBA_PortStatistics.LinkFailureCount.
PrimitiveSeqProtocolErrCount (added)	Mandatory	
LossOfSignalCounter (added)	Mandatory	Maps to HBA_PortStatistics.LossOfSignalCount.
InvalidTransmissionWords (added)	Mandatory	Maps to HBA_PortStatistics.InvalidTxWordCount.
StatisticTime (added)	Optional	Time last measurement was taken.
LIPCount (added)	Mandatory	
NOSCount (added)	Mandatory	
ErrorFrames (added)	Mandatory	
DumpedFrames (added)	Mandatory	
LossOfSyncCounter (added)	Mandatory	Maps to HBA_PortStatistics.LossOfSynchCount.

18.6.7 CIM_HostedAccessPoint (Initiator)

Associates system to initiator protocol endpoints. The class definition specializes the CIM_HostedAccessPoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 157 describes class CIM_HostedAccessPoint (Initiator).

Table 157 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint(Initiator).

18.6.8 CIM_HostedAccessPoint (Target)

Associates system to optional remote protocol endpoints. The class definition specializes the CIM_HostedAccessPoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 158 describes class CIM_HostedAccessPoint (Target).

Table 158 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint(Target).

18.6.9 CIM_HostedCollection (Connectivity Collection)

Associates the ConnectivityCollection to the hosting System.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 159 describes class CIM_HostedCollection (Connectivity Collection).

Table 159 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Dependent	Mandatory	Reference to ConnectivityCollection.
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.

18.6.10 CIM_MemberOfCollection (Connectivity Collection)

Associates SCSIProtocolEndpoints to the ConnectivityCollection. The class definition specializes the CIM_MemberOfCollection definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 160 describes class CIM_MemberOfCollection (Connectivity Collection).

Table 160 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Member (overridden)	Mandatory	Reference to target or initiator SCSIProtocolEndpoint.
Collection	Mandatory	Reference to ConnectivityCollection.

18.6.11 CIM_ProtocolControllerForPort

Deprecated. Associates SCSIProtocolController to FCPort.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 161 describes class CIM_ProtocolControllerForPort.

Table 161 - SMI Referenced Properties/Methods for CIM_ProtocolControllerForPort

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to SCSIProtocolController.
Dependent	Mandatory	Reference to initiator FCPort.

18.6.12 CIM_SCSIInitiatorTargetLogicalUnitPath

Represents a path between a SCSI initiator, target, and logical unit.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 162 describes class CIM_SCSIInitiatorTargetLogicalUnitPath.

Table 162 - SMI Referenced Properties/Methods for CIM_SCSIInitiatorTargetLogicalUnitPath

Properties	Requirement	Description and Notes
LogicalUnit	Mandatory	Reference to StorageExtent in Disk Drive Lite Profile or MediaAccessDevice in Media Access Device Profile.
Initiator	Mandatory	Reference to SCSIProtocolEndpoint(Initiator).
Target	Mandatory	Reference to SCSIProtocolEndpoint(Target).

18.6.13 CIM_SCSIProtocolController

Deprecated. Represents a SCSI logical unit inventory.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 163 describes class CIM_SCSIProtocolController.

Table 163 - SMI Referenced Properties/Methods for CIM_SCSIProtocolController

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	Opaque identifier.
ElementName	Optional	
OperationalStatus	Optional	
MaxUnitsControlled	Optional	

18.6.14 CIM_SCSIProtocolEndpoint (Initiator)

Represents support for the SCSI command set. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 164 describes class CIM_SCSIProtocolEndpoint (Initiator).

Table 164 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Initiator)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	See <i>Information Technology – Storage Management, Part 2 Common Architecture</i> 8.6.3 Standard Formats for Port Names.
ProtocolIFType	Mandatory	Shall be 1 (Other).
OtherTypeDescription (overridden)	Mandatory	Shall be the string 'SCSI'.
ConnectionType (added)	Mandatory	Shall be 2 (Fibre Channel).
Role (added)	Mandatory	Shall be 2 (Initiator) or 4 (Both Initiator and Target).

18.6.15 CIM_SCSIProtocolEndpoint (Target)

Models remote ports - target devices and possibly other initiators. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 165 describes class CIM_SCSIProtocolEndpoint (Target).

Table 165 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Target)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType	Mandatory	The values in MOFs map to IETF values and exclude storage. Shall be 1 (Other) and set OtherTypeDescription appropriately.
OtherTypeDescription (overridden)	Mandatory	Shall be the string 'SCSI'.
Role (added)	Mandatory	Should be set appropriately by the instrumentation. If not known, use 0 (Unknown).
ConnectionType (added)	Mandatory	Shall be 8 (FC).

18.6.16 CIM_SystemDevice (Initiator Ports)

Associates system to initiator ports. The class definition specializes the CIM_SystemDevice definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 166 describes class CIM_SystemDevice (Initiator Ports).

Table 166 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to ComputerSystem.
PartComponent (overridden)	Mandatory	Reference to FCPort.

STABLE

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

EXPERIMENTAL

19 SAS Initiator Ports Profile

19.1 Synopsis

Profile Name: SAS Initiator Ports (Component Profile)

Version: 1.4.0

Organization: SNIA

CIM Schema Version: 2.11.0

Related Profiles for SAS Initiator Ports: Not defined in this International Standard.

Specializes: Generic Initiator Port Profile

Central Class: CIM_SASPort

Scoping Class: a CIM_System in a separate autonomous profile

The SAS Initiator Port Profile models the management of a Serial Attached SCSI port that initiates commands to devices.

19.2 Description

19.2.1 General

The SAS Initiator Port Profile defines the model Serial Attached SCSI (SAS) ports. A typical instance diagram is provided in Figure 26.

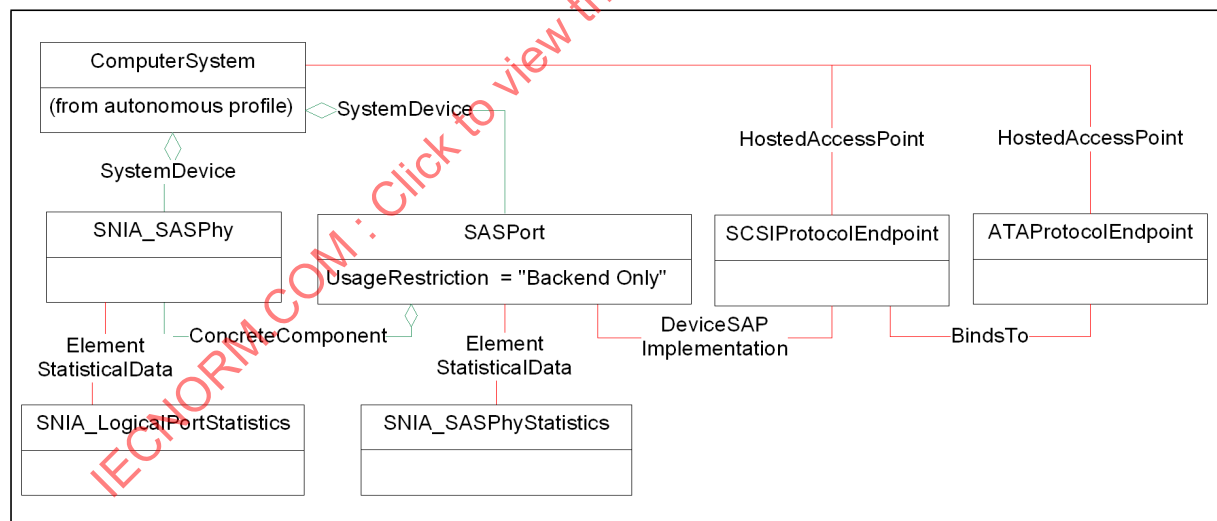


Figure 26 - SAS Initiator Port Model

The SASPhy class represents a SAS PHY. A SAS Port may have multiple associated PHYs; generally, all the PHYs are connected to the same target or expander and provide additional bandwidth.

SASPort represents a SAS initiator port which is an aggregation of SASPHY instances.

SNIA_SASPhyStatistics is optional and may be associated to SASPhy to hold PHY error statistics.

SNIA_LogicalPortStatistics is optional and may be associated to SASPort to hold port I/O statistics.

ATAProtocolEndpoint associated to SCSIProtocolEndpoint using BindsTo shall be used to represent support for ATA (SATA) tunneled over SCSI.

19.2.2 Health and Fault Management Considerations

Table 167 summarizes the Health and Fault Management issues that are unique to this profile.

Table 167 - SASPort OperationalStatus

OperationalStatus	Description
OK	Port is online
Error	Port has a failure
Stopped	Port is disabled
InService	Port is in Self Test
Unknown	

19.3 Methods of the profile

Not defined in this International Standard.

19.4 Client Considerations and Recipes

Not defined in this International Standard.

19.5 CIM Elements

19.5.1 General

Table 168 describes the CIM elements for SAS Initiator Ports.

Table 168 - CIM Elements for SAS Initiator Ports

Element Name	Requirement	Description
19.5.2 CIM_ATAProtocolEndpoint (Initiator)	Optional	Initiator ATA endpoints.
19.5.3 CIM_BindsTo	Optional	Associates SCSIProtocolEndpoint and ATAProtocolEndpoint.
19.5.4 CIM_ConcreteComponent	Mandatory	Associates SASPort and SASPHY.
19.5.5 CIM_ConnectivityCollection	Optional	Represents a collection of connected ProtocolEndpoints.
19.5.6 CIM_DeviceSAPImplementation	Mandatory	Connects Initiator SASLogicalPort and SCSIProtocolEndpoint.
19.5.7 CIM_ElementStatisticalData (PHY Statistics)	Optional	Associates SASPort and SASPhyStatistics.
19.5.8 CIM_ElementStatisticalData (Port Statistics)	Optional	Connects LogicalPort and LogicalPortStatistics.
19.5.9 CIM_HostedAccessPoint (Initiator)	Mandatory	Associates system to initiator protocol endpoints.
19.5.10 CIM_HostedAccessPoint (Target)	Optional	Associates system to optional remote protocol endpoints.
19.5.11 CIM_HostedCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Associates the ConnectivityCollection to the hosting System.

Table 168 - CIM Elements for SAS Initiator Ports

Element Name	Requirement	Description
19.5.12 CIM_MemberOfCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Associates ProtocolEndpoints to the ConnectivityCollection.
19.5.13 CIM_SASPort	Mandatory	Represents the logical aspects of the physical port and may have multiple associated protocols.
19.5.14 CIM_SCSIInitiatorTargetLogicalUnitPath	Optional	Represents a path between a SCSI initiator, target, and logical unit.
19.5.15 CIM_SCSIProtocolEndpoint (Initiator)	Mandatory	Represents support for the SCSI command set.
19.5.16 CIM_SCSIProtocolEndpoint (Target)	Optional	Models remote ports - target devices and possibly other initiators.
19.5.17 CIM_SystemDevice (Initiator PHY)	Mandatory	Associates system to initiator SAS PHYs.
19.5.18 CIM_SystemDevice (Initiator Ports)	Mandatory	Associates system to initiator ports.
19.5.19 SNIA_LogicalPortStatistics	Optional	Statistics for a port.
19.5.20 SNIA_SASPHY	Mandatory	A PHY on a SAS HBA, Expander, or device.
19.5.21 SNIA_SASPhyStatistics	Optional	Statistics for a SAS PHY.

19.5.2 CIM_ATAProtocolEndpoint (Initiator)

Initiator ATA endpoints.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 169 describes class CIM_ATAProtocolEndpoint (Initiator).

Table 169 - SMI Referenced Properties/Methods for CIM_ATAProtocolEndpoint (Initiator)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	See <i>Information Technology – Storage Management, Part 2 Common Architecture</i> 8.6.3 Standard Formats for Port Names.
ProtocolIFTType	Mandatory	Shall be 1 (Other).
OtherTypeDescription	Mandatory	Shall be 'ATA'.
ConnectionType	Mandatory	Shall be 3 (SATA).
Role	Mandatory	Shall be 3 (Target).

19.5.3 CIM_BindsTo

Associates SCSIProtocolEndpoint and ATAProtocolEndpoint.

Created By: Static
 Modified By: Static
 Deleted By: Static
 Requirement: Optional

Table 170 describes class CIM_BindsTo.

Table 170 - SMI Referenced Properties/Methods for CIM_BindsTo

Properties	Requirement	Description and Notes
Dependent	Mandatory	Reference to ATAProtocolEndpoint.
Antecedent	Mandatory	Reference to SCSIProtocolEndpoint.

19.5.4 CIM_ConcreteComponent

Associates SASPort and SASPHY.

Created By: Static
 Modified By: Static
 Deleted By: Static
 Requirement: Mandatory

Table 171 describes class CIM_ConcreteComponent.

Table 171 - SMI Referenced Properties/Methods for CIM_ConcreteComponent

Properties	Requirement	Description and Notes
PartComponent	Mandatory	Reference to SASPHY.
GroupComponent	Mandatory	Reference to SASPort.

19.5.5 CIM_ConnectivityCollection

Represents a collection of connected ProtocolEndpoints.

Created By: Static
 Modified By: Static
 Deleted By: Static
 Requirement: Optional

Table 172 describes class CIM_ConnectivityCollection.

Table 172 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection

Properties	Requirement	Description and Notes
InstanceID	Mandatory	

19.5.6 CIM_DeviceSAPImplementation

Connects Initiator SASLogicalPort and SCSIProtocolEndpoint. The class definition specializes the CIM_DeviceSAPImplementation definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 173 describes class CIM_DeviceSAPImplementation.

Table 173 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Requirement	Description and Notes
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint(Initiator).
Antecedent (overridden)	Mandatory	Reference to SASPort.

19.5.7 CIM_ElementStatisticalData (PHY Statistics)

Associates SASPort and SASPhyStatistics.

Requirement: Optional

Table 174 describes class CIM_ElementStatisticalData (PHY Statistics).

Table 174 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (PHY Statistics)

Properties	Requirement	Description and Notes
ManagedElement	Mandatory	Reference to SASPort.
Stats	Mandatory	Reference to SASPhyStatistics.

19.5.8 CIM_ElementStatisticalData (Port Statistics)

Connects LogicalPort and LogicalPortStatistics.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 175 describes class CIM_ElementStatisticalData (Port Statistics).

Table 175 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)

Properties	Requirement	Description and Notes
ManagedElement	Mandatory	Reference to LogicalPort.
Stats	Mandatory	Reference to LogicalPortStatistics.

19.5.9 CIM_HostedAccessPoint (Initiator)

Associates system to initiator protocol endpoints. The class definition specializes the CIM_HostedAccessPoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 176 describes class CIM_HostedAccessPoint (Initiator).

Table 176 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint(Initiator).

19.5.10 CIM_HostedAccessPoint (Target)

Associates system to optional remote protocol endpoints. The class definition specializes the CIM_HostedAccessPoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 177 describes class CIM_HostedAccessPoint (Target).

Table 177 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint(Target).

19.5.11 CIM_HostedCollection (Connectivity Collection)

Associates the ConnectivityCollection to the hosting System.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 178 describes class CIM_HostedCollection (Connectivity Collection).

Table 178 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Dependent	Mandatory	Reference to ConnectivityCollection.
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.

19.5.12 CIM_MemberOfCollection (Connectivity Collection)

Associates ProtocolEndpoints to the ConnectivityCollection. The class definition specializes the CIM_MemberOfCollection definition in the Generic Initiator Ports profile. Properties or

methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 179 describes class CIM_MemberOfCollection (Connectivity Collection).

Table 179 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Member (overridden)	Mandatory	Reference to SCSIProtocolEndpoint.
Collection	Mandatory	Reference to ConnectivityCollection.

19.5.13 CIM_SASPort

Represents the logical aspects of the physical port and may have multiple associated protocols. The class definition specializes the CIM_LogicalPort definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 180 describes class CIM_SASPort.

Table 180 - SMI Referenced Properties/Methods for CIM_SASPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus (overridden)	Mandatory	Shall be 0 (Unknown), 2 (OK), 6 (Error), 10 (Stopped), or 11 (In Service).
UsageRestriction (overridden)	Mandatory	Shall be 3 (Back-end Only).
PortType (overridden)	Mandatory	Shall be 94 (SAS).
PermanentAddress (added)	Mandatory	SAS Address. Shall be 16 un-separated upper case hex digits. See <i>Information Technology – Storage Management, Part 2 Common Architecture</i> 8.6.3 Standard Formats for Port Names.

19.5.14 CIM_SCSIInitiatorTargetLogicalUnitPath

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 181 describes class CIM_SCSIInitiatorTargetLogicalUnitPath.

Table 181 - SMI Referenced Properties/Methods for CIM_SCSIInitiatorTargetLogicalUnitPath

Properties	Requirement	Description and Notes
LogicalUnit	Mandatory	Reference to StorageExtent in Disk Drive Lite Profile or MediaAccessDevice in Media Access Device Profile.
Target	Mandatory	Reference to SCSIProtocolEndpoint(Target).
Initiator	Mandatory	Reference to SCSIProtocolEndpoint(Initiator).

19.5.15 CIM_SCSIProtocolEndpoint (Initiator)

Represents support for the SCSI command set. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 182 describes class CIM_SCSIProtocolEndpoint (Initiator).

Table 182 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Initiator)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	See <i>Information Technology – Storage Management, Part 2 Common Architecture</i> 8.6.3 Standard Formats for Port Names.
ProtocolType	Mandatory	Shall be 1 (Other).
OtherTypeDescription (overridden)	Mandatory	Shall be the string 'SCSI'.
ConnectionType (added)	Mandatory	Shall be 8 (SAS).
Role (added)	Mandatory	Shall be 2 (Initiator).

19.5.16 CIM_SCSIProtocolEndpoint (Target)

Models remote ports - target devices and possibly other initiators. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 183 describes class CIM_SCSIProtocolEndpoint (Target).

Table 183 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Target)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolType	Mandatory	The values in MOFs map to IETF values and exclude storage. Shall be 1 (Other) and set OtherTypeDescription appropriately.
OtherTypeDescription (overridden)	Mandatory	Shall be the string 'SCSI'.
Role (added)	Mandatory	SCSI target or initiator role. Should be set appropriately by the instrumentation. If not know, use 0 (Unknown).
ConnectionType (added)	Mandatory	Shall be 8 (SAS).

19.5.17 CIM_SystemDevice (Initiator PHY)

Associates system to initiator SAS PHYs.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 184 describes class CIM_SystemDevice (Initiator PHY).

Table 184 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator PHY)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to ComputerSystem.
PartComponent	Mandatory	Reference to SASPHY.

19.5.18 CIM_SystemDevice (Initiator Ports)

Associates system to initiator ports. The class definition specializes the CIM_SystemDevice definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 185 describes class CIM_SystemDevice (Initiator Ports).

Table 185 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to ComputerSystem.
PartComponent (overridden)	Mandatory	Reference to back-end (initiator) SASPorts.

19.5.19 SNIA_LogicalPortStatistics

Statistics for a port.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 186 describes class SNIA_LogicalPortStatistics.

Table 186 - SMI Referenced Properties/Methods for SNIA_LogicalPortStatistics

Properties	Requirement	Description and Notes
ElementName	Mandatory	
InstanceID	Mandatory	
BytesTransmitted	Mandatory	
BytesReceived	Mandatory	
Packets Transmitted	Mandatory	
Packets Received	Mandatory	

19.5.20 SNIA_SASPHY

A PHY on a SAS HBA, Expander, or device.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 187 describes class SNIA_SASPHY.

Table 187 - SMI Referenced Properties/Methods for SNIA_SASPHY

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
HardwareMinimumPhysicalLinkRate	Mandatory	
HardwareMaximumPhysicalLinkRate	Mandatory	

Table 187 - SMI Referenced Properties/Methods for SNIA_SASPHY

Properties	Requirement	Description and Notes
ProgrammedMinimumPhysicalLinkRate	Mandatory	
ProgrammedMaximumPhysicalLinkRate	Mandatory	
NegotiatedPhysicalLinkRate	Mandatory	

19.5.21 SNIA_SASPhyStatistics

Statistics for a SAS PHY.

Requirement: Optional

Table 188 describes class SNIA_SASPhyStatistics.

Table 188 - SMI Referenced Properties/Methods for SNIA_SASPhyStatistics

Properties	Requirement	Description and Notes
InvalidDwordCount	Mandatory	
RunningDisparityErrorCount	Mandatory	
LossOfDwordSyncCount	Mandatory	
ResetProblemCount	Mandatory	

EXPERIMENTAL

EXPERIMENTAL

20 ATA Initiator Ports Profile

20.1 Synopsis

Profile Name: ATA Initiator Ports (Component Profile)

Version: 1.4.0

Organization: SNIA

CIM Schema Version: 2.13.1

Related Profiles for ATA Initiator Ports: Not defined in this International Standard.

Specializes: Generic Initiator Port Profile

Central Class: CIM_ATAPort

Scoping Class: a CIM_System in a separate autonomous profile

The ATA Initiator Ports Profile models the management of a PATA or SATA port that initiates commands to devices.

20.2 Description

The ATA Initiator Port Profile describes the model for Parallel or Serial ATA Ports with optional attached drives.

20.3 Implementation

20.3.1 General

The port is modeled as ATAPort (with PortType set to ATA for PATA ports or SATA) and ATAProtocolEndpoint associated by DeviceSAPImplementation. Attached drives are optionally modeled as subclasses of LogicalDevice (e.g., StorageVolume, TapeDrive) which are associated via SAPAvailableToElement to ATAProtocolEndpoint.

Figure 27 shows a class diagram for this profile.

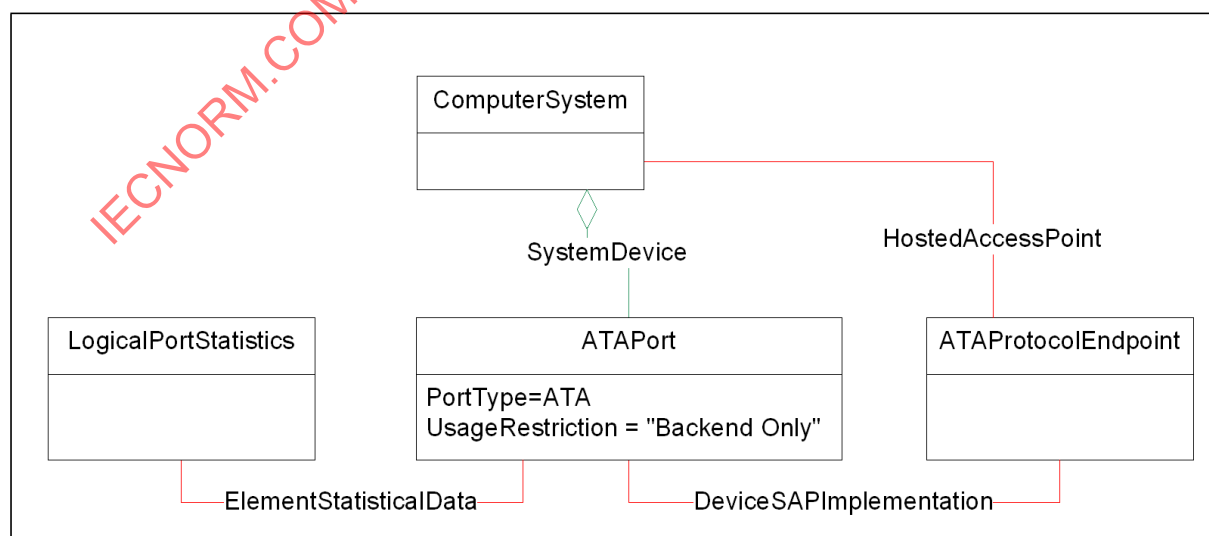


Figure 27 - ATA Initiator Port Class Diagram

20.3.2 Health and Fault Management Consideration

Table 189 summarizes the Health and Fault Management considerations that are specific to this profile.

Table 189 - ATAPort OperationalStatus

OperationalStatus	Description
OK	Port is online
Error	Port has a failure
Stopped	Port is disabled
InService	Port is in Self Test
Unknown	

20.3.3 Cascading Considerations

Not defined in this International Standard.

20.4 Methods of the Profile

20.4.1 Extrinsic Methods of the Profile

None.

20.4.2 Intrinsic Methods of this Profile

The profile supports read methods and association traversal. Specifically, the list of intrinsic operations supported are as follows:

- GetInstance
- Associators
- AssociatorNames
- References
- ReferenceNames
- EnumerateInstances
- EnumerateInstanceNames

20.5 Client Considerations and Recipes

None.

20.6 CIM Elements

20.6.1 General

Table 190 describes the CIM elements for ATA Initiator Ports.

Table 190 - CIM Elements for ATA Initiator Ports

Element Name	Requirement	Description
20.6.2 CIM_ATAInitiatorTargetLogicalUnitPath	Optional	Represents a path between an ATA initiator, target, and logical unit.
20.6.3 CIM_ATAPort	Mandatory	Represents the logical aspects of the physical port and may have multiple associated protocols.
20.6.4 CIM_ATAProtocolEndpoint (Initiator)	Mandatory	ProtocolEndpoints associated to initiator ports.
20.6.5 CIM_ATAProtocolEndpoint (Target)	Mandatory	Models remote ports - target devices and possibly other initiators.
20.6.6 CIM_ConnectivityCollection	Mandatory	Represents a collection of connected ATAProtocolEndpoints.
20.6.7 CIM_DeviceSAPImplementation	Mandatory	Connects Initiator ATALogicalPort and ATAProtocolEndpoint.
20.6.8 CIM_ElementStatisticalData (Port Statistics)	Mandatory	Connects ATAPort and LogicalPortStatistics.
20.6.9 CIM_HostedAccessPoint (Initiator)	Mandatory	Associates system to initiator protocol endpoints.
20.6.10 CIM_HostedAccessPoint (Target)	Optional	Associates system to optional remote protocol endpoints.
20.6.11 CIM_HostedCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Associates the ConnectivityCollection to the hosting System.
20.6.12 CIM_MemberOfCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Represents a collection of connected ATAProtocolEndpoints.
20.6.13 CIM_SystemDevice (Initiator Ports)	Mandatory	Associates system to initiator ports.
20.6.14 SNIA_LogicalPortStatistics	Optional	Statistics for a port.

20.6.2 CIM_ATAInitiatorTargetLogicalUnitPath

Represents a path between an ATA initiator, target, and logical unit.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 191 describes class CIM_ATAInitiatorTargetLogicalUnitPath.

Table 191 - SMI Referenced Properties/Methods for CIM_ATAInitiatorTargetLogicalUnitPath

Properties	Requirement	Description and Notes
LogicalUnit	Mandatory	Reference to StorageExtent in Disk Drive Lite Profile or MediaAccessDevice in Media Access Device Profile.
Target	Mandatory	Reference to ATAProtocolEndpoint(Target).
Initiator	Mandatory	Reference to ATAProtocolEndpoint(Initiator).

20.6.3 CIM_ATAPort

Represents the logical aspects of the physical port and may have multiple associated protocols. The class definition specializes the CIM_LogicalPort definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 192 describes class CIM_ATAPort.

Table 192 - SMI Referenced Properties/Methods for CIM_ATAPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus	Mandatory	
UsageRestriction	Mandatory	Shall be 3 for ports restricted to back-end (initiator) only or 4 if the port is unrestricted.
PortType (overridden)	Mandatory	Shall be 91(ATA), 92(SATA) or 93(SATA2).
Name (added)	Mandatory	See <i>Information Technology – Storage Management, Part 2 Common Architecture</i> 8.6.3 Standard Formats for Port Names.

20.6.4 CIM_ATAProtocolEndpoint (Initiator)

ProtocolEndpoints associated to initiator ports. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 193 describes class CIM_ATAProtocolEndpoint (Initiator).

Table 193 - SMI Referenced Properties/Methods for CIM_ATAProtocolEndpoint (Initiator)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name (overridden)	Mandatory	See <i>Information Technology – Storage Management, Part 2 Common Architecture</i> 8.6.3 Standard Formats for Port Names.
ProtocolIFType	Mandatory	Shall be 1 (Other).
OtherTypeDescription (overridden)	Mandatory	Shall be 'ATA'.
Role (added)	Mandatory	Shall be 2 (Initiator).
ConnectionType (added)	Mandatory	Shall be 2 (ATA for PATA ports) or 3 (SATA).

20.6.5 CIM_ATAProtocolEndpoint (Target)

Models remote ports - target devices and possibly other initiators. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 194 describes class CIM_ATAProtocolEndpoint (Target).

Table 194 - SMI Referenced Properties/Methods for CIM_ATAProtocolEndpoint (Target)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType	Mandatory	The values in MOFs map to IETF values and exclude storage. Shall be 1 (Other) and set OtherTypeDescription appropriately.
OtherTypeDescription (overridden)	Mandatory	Shall be 'ATA'.
Role (added)	Mandatory	Should be set appropriately by the instrumentation. If unknown, use 0 (Unknown).
ConnectionType (added)	Mandatory	Shall be 2 (ATA for PATA ports) or 3 (SATA).

20.6.6 CIM_ConnectivityCollection

Represents a collection of connected ATAProtocolEndpoints. The class definition specializes the CIM_ConnectivityCollection definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 195 describes class CIM_ConnectivityCollection.

Table 195 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection

Properties	Requirement	Description and Notes
InstanceID	Mandatory	

20.6.7 CIM_DeviceSAPImplementation

Connects Initiator ATALogicalPort and ATAProtocolEndpoint. The class definition specializes the CIM_DeviceSAPImplementation definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 196 describes class CIM_DeviceSAPImplementation.

Table 196 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Requirement	Description and Notes
Dependent (overridden)	Mandatory	Reference to ATAProtocolEndpoint(Initiator).
Antecedent (overridden)	Mandatory	Reference to ATAPort.

20.6.8 CIM_ElementStatisticalData (Port Statistics)

Connects ATAPort and LogicalPortStatistics. The class definition specializes the CIM_ElementStatisticalData definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 197 describes class CIM_ElementStatisticalData (Port Statistics).

Table 197 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)

Properties	Requirement	Description and Notes
ManagedElement (overridden)	Mandatory	Reference to ATAPort.
Stats	Mandatory	Reference to LogicalPortStatistics.

20.6.9 CIM_HostedAccessPoint (Initiator)

Associates system to initiator protocol endpoints. The class definition specializes the CIM_HostedAccessPoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 198 describes class CIM_HostedAccessPoint (Initiator).

Table 198 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent (overridden)	Mandatory	Reference to ATAProtocolEndpoint(Initiator).

20.6.10 CIM_HostedAccessPoint (Target)

Associates system to optional remote protocol endpoints. The class definition specializes the CIM_HostedAccessPoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 199 describes class CIM_HostedAccessPoint (Target).

Table 199 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent (overridden)	Mandatory	Reference to ATAProtocolEndpoint(Target).

20.6.11 CIM_HostedCollection (Connectivity Collection)

Associates the ConnectivityCollection to the hosting System.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 200 describes class CIM_HostedCollection (Connectivity Collection).

Table 200 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Dependent	Mandatory	Reference to ConnectivityCollection.
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.

20.6.12 CIM_MemberOfCollection (Connectivity Collection)

Represents a collection of connected ATAProtocolEndpoints. The class definition specializes the CIM_MemberOfCollection definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 201 describes class CIM_MemberOfCollection (Connectivity Collection).

Table 201 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Member (overridden)	Mandatory	Reference to ATAProtocolEndpoint(Initiator or Target).
Collection	Mandatory	Reference to ConnectivityCollection.

20.6.13 CIM_SystemDevice (Initiator Ports)

Associates system to initiator ports. The class definition specializes the CIM_SystemDevice definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 202 describes class CIM_SystemDevice (Initiator Ports).

Table 202 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to ComputerSystem.
PartComponent (overridden)	Mandatory	Reference to ATAPort(Initiator).

20.6.14 SNIA_LogicalPortStatistics

Statistics for a port.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 203 describes class SNIA_LogicalPortStatistics.

Table 203 - SMI Referenced Properties/Methods for SNIA_LogicalPortStatistics

Properties	Requirement	Description and Notes
ElementName	Mandatory	
InstanceID	Mandatory	
BytesTransmitted	Mandatory	
BytesReceived	Mandatory	
Packets Transmitted	Mandatory	
Packets Received	Mandatory	

EXPERIMENTAL

EXPERIMENTAL

21 FC-SB-x Initiator Ports Profile

21.1 Synopsis

Profile Name: SB Initiator Ports (Component Profile)

Version: 1.4.0

Organization: SNIA

CIM Schema Version: 2.13.0

Table 204 describes the related profiles for SB Initiator Ports.

Table 204 - Related Profiles for SB Initiator Ports

Profile Name	Organization	Version	Requirement	Description
Indication	SNIA	1.5.0	Mandatory	

The FC-SB-x Initiator Ports Profile models initiator ports that support the FC-SB-x protocol.

21.2 Description

The FC-SB-x Initiator Ports Profile models initiator ports that support the FC-SB-x protocol.

21.3 Implementation

21.3.1 General

Figure 28 is an example of a single initiator port. The instance diagram shows a disk (LogicalDevice in the diagram would be subclassed as something like StorageExtent) in an array, connected by a Fibre Channel port. The full model for the disk is shown in the Disk Drive Lite Profile. SBProtocolController is not generally used in initiator contexts. It is included here to be compatible with SMI-S 1.0 clients.

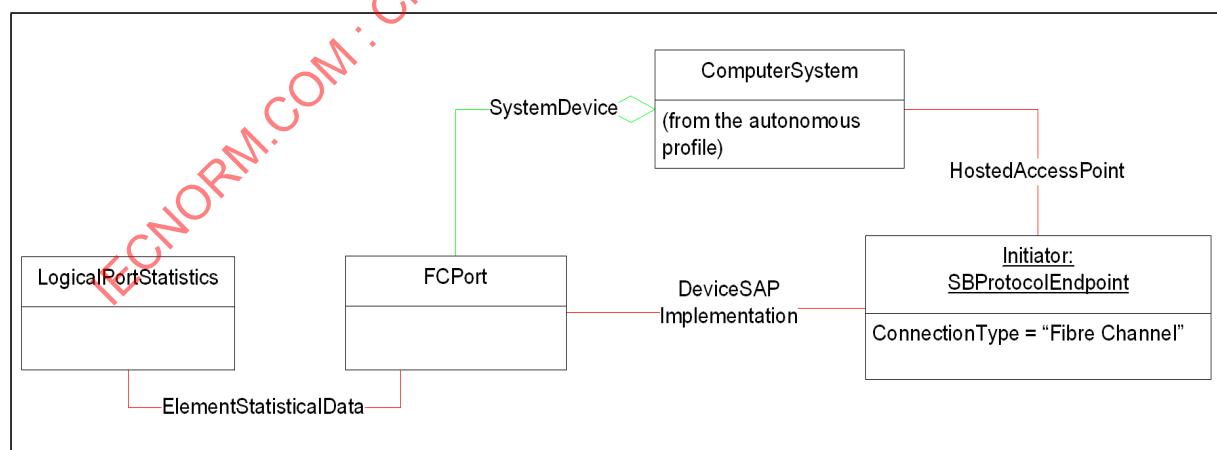


Figure 28 - Fibre Channel Initiator Instance Diagram

21.3.2 Health and Fault Management Considerations

Table 205 summarizes the Health and Fault Management considerations specific to this profile.

Table 205 - FCPort OperationalStatus

OperationalStatus	Description
OK	Port is online
Error	Port has a failure
Stopped	Port is disabled
InService	Port is in Self Test
Unknown	

21.3.3 Cascading Considerations

Not defined in this International Standard.

21.4 Methods

21.4.1 Extrinsic Methods of the Profile

None.

21.4.2 Intrinsic Methods of this Profile

The profile supports read methods and association traversal. Specifically, the list of intrinsic operations supported are as follows:

- GetInstance
- Associators
- AssociatorNames
- References
- ReferenceNames
- EnumerateInstances
- EnumerateInstanceNames

21.5 Client Considerations and Recipes

None.

21.6 CIM Elements

21.6.1 General

Table 206 describes the CIM elements for SB Initiator Ports.

Table 206 - CIM Elements for SB Initiator Ports

Element Name	Requirement	Description
21.6.2 CIM_ConnectivityCollection	Mandatory	Represents a collection of connected SBProtocolEndpoints.
21.6.3 CIM_DeviceSAPImplementation	Mandatory	Connects Initiator SBLogicalPort and SBProtocolEndpoint.
21.6.4 CIM_ElementStatisticalData (Port Statistics)	Mandatory	Connects SBPort and LogicalPortStatistics.
21.6.5 CIM_FCPort	Mandatory	Represents the logical aspects of the physical port and may have multiple associated protocols.
21.6.6 CIM_HostedAccessPoint (Initiator)	Mandatory	Associates system to initiator protocol endpoints.
21.6.7 CIM_HostedAccessPoint (Target)	Optional	Associates system to optional remote protocol endpoints.
21.6.8 CIM_HostedCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Associates the ConnectivityCollection to the hosting System.
21.6.9 CIM_MemberOfCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Represents a collection of connected SBProtocolEndpoints.
21.6.10 CIM_SystemDevice (Initiator Ports)	Mandatory	Associates system to initiator ports.
21.6.11 SNIA_LogicalPortStatistics	Optional	Statistics for a port.
21.6.12 SNIA_SBInitiatorTargetLogicalUnitPath	Optional	
21.6.13 SNIA_SBProtocolEndpoint (Initiator)	Mandatory	Represents a protocol (command set) supported by the port. The appropriate subclass (SCSIProtocolEndpoint, ATAProtocolEndpoint, SBProtocolEndpoint) should be used in initiator port specialized profiles.
21.6.14 SNIA_SBProtocolEndpoint (Target)	Mandatory	Target or non-local ProtocolEndpoint.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_FCPort	Optional	Create FCPort.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.CIM_FCPort::OperationalStatus <> PreviousInstance.CIM_FCPort::OperationalStatus	Optional	CQL -Modify FCPort.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_FCPort	Optional	Delete FCPort.

21.6.2 CIM_ConnectivityCollection

Represents a collection of connected SBProtocolEndpoints. The class definition specializes the CIM_ConnectivityCollection definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 207 describes class CIM_ConnectivityCollection.

Table 207 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection

Properties	Requirement	Description and Notes
InstanceID	Mandatory	

21.6.3 CIM_DeviceSAPImplementation

Connects Initiator SBLogicalPort and SBProtocolEndpoint. The class definition specializes the CIM_DeviceSAPImplementation definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 208 describes class CIM_DeviceSAPImplementation.

Table 208 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Requirement	Description and Notes
Dependent (overridden)	Mandatory	Reference to SBProtocolEndpoint(Initiator).
Antecedent (overridden)	Mandatory	Reference to FCPort.

21.6.4 CIM_ElementStatisticalData (Port Statistics)

Connects SBPort and LogicalPortStatistics. The class definition specializes the CIM_ElementStatisticalData definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 209 describes class CIM_ElementStatisticalData (Port Statistics).

Table 209 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)

Properties	Requirement	Description and Notes
ManagedElement (overridden)	Mandatory	Reference to FCPort.
Stats	Mandatory	Reference to LogicalPortStatistics.

21.6.5 CIM_FCPort

Represents the logical aspects of the physical port and may have multiple associated protocols. The class definition specializes the CIM_LogicalPort definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 210 describes class CIM_FCPort.

Table 210 - SMI Referenced Properties/Methods for CIM_FCPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus	Mandatory	
UsageRestriction (overridden)	Mandatory	Shall be 3 for ports restricted to Back-end only or 4 if the port is unrestricted.
PortType (overridden)	Mandatory	Shall be 0 1 10 11 12 13 14 15 16 17 18 (Unknown or Other or N or NL or F/NL or Nx or E or F or FL or B or G).
ElementName (added)	Mandatory	Port Symbolic Name.
Speed (added)	Mandatory	
MaxSpeed (added)	Mandatory	Port Supported Speed from HBA API.
PortNumber (added)	Optional	
PermanentAddress (added)	Optional	Port WWN. PermanentAddress is optional when used as a backend port in a device. This may be overridden in profiles that use this profile. See <i>Information Technology – Storage Management, Part 2 Common Architecture</i> 8.6.3 Standard Formats for Port Names.
NetworkAddresses (added)	Optional	For Fibre Channel end device ports, the Fibre Channel ID.
SupportedCOS (added)	Optional	
ActiveCOS (added)	Optional	
SupportedFC4Types (added)	Optional	
ActiveFC4Types (added)	Optional	
LinkTechnology (added)	Mandatory	
SupportedMaximumTransmissionUnit (added)	Mandatory	
ActiveMaximumTransmissionUnit (added)	Optional	

21.6.6 CIM_HostedAccessPoint (Initiator)

Associates system to initiator protocol endpoints.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 211 describes class CIM_HostedAccessPoint (Initiator).

Table 211 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent	Mandatory	Reference to ProtocolEndpoint(Initiator).

21.6.7 CIM_HostedAccessPoint (Target)

Associates system to optional remote protocol endpoints.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 212 describes class CIM_HostedAccessPoint (Target).

Table 212 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent	Mandatory	Reference to ProtocolEndpoint(Target).

21.6.8 CIM_HostedCollection (Connectivity Collection)

Associates the ConnectivityCollection to the hosting System.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 213 describes class CIM_HostedCollection (Connectivity Collection).

Table 213 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Dependent	Mandatory	Reference to ConnectivityCollection.
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.

21.6.9 CIM_MemberOfCollection (Connectivity Collection)

Represents a collection of connected SBProtocolEndpoints. The class definition specializes the CIM_MemberOfCollection definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 214 describes class CIM_MemberOfCollection (Connectivity Collection).

Table 214 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Member (overridden)	Mandatory	Reference to SBProtocolEndpoint(Initiator or Target).
Collection	Mandatory	Reference to ConnectivityCollection.

21.6.10 CIM_SystemDevice (Initiator Ports)

Associates system to initiator ports.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 215 describes class CIM_SystemDevice (Initiator Ports).

Table 215 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to ComputerSystem.
PartComponent	Mandatory	Reference to LogicalPort.

21.6.11 SNIA_LogicalPortStatistics

Statistics for a port.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 216 describes class SNIA_LogicalPortStatistics.

Table 216 - SMI Referenced Properties/Methods for SNIA_LogicalPortStatistics

Properties	Requirement	Description and Notes
ElementName	Mandatory	
InstanceID	Mandatory	
BytesTransmitted	Mandatory	
BytesReceived	Mandatory	
PacketsTransmitted	Mandatory	
PacketsReceived	Mandatory	

21.6.12 SNIA_SBInitiatorTargetLogicalUnitPath

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 217 describes class SNIA_SBInitiatorTargetLogicalUnitPath.

Table 217 - SMI Referenced Properties/Methods for SNIA_SBInitiatorTargetLogicalUnitPath

Properties	Requirement	Description and Notes
UsePreferredPath	Optional	Boolean indicating whether preferred path processing is required.
PreferredPath	Optional	Boolean indicating whether this is a preferred path.
PathGroupState	Optional	One of 0(Unknown), 2(Path grouping not supported), 3(Reset), 4(Grouped), or 5(Ungrouped).
PathGroupMode	Optional	One of 0(Unknown), 2(None), 3(Single path), or 4(Multipath). Single path and multipath only valid if PathGroupState is grouped.
PathGroupID	Optional	String containing the ID from the OS, only valid if PathGroupState is Grouped.
LogicalUnit	Mandatory	Reference to StorageExtent in Disk Drive Lite Profile or MediaAccessDevice in Media Access Device Profile.
Target	Mandatory	Reference to SCSIProtocolEndpoint(Target).
Initiator	Mandatory	Reference to SCSIProtocolEndpoint(Initiator).

21.6.13 SNIA_SBProtocolEndpoint (Initiator)

Represents a protocol (command set) supported by the port. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 218 describes class SNIA_SBProtocolEndpoint (Initiator).

Table 218 - SMI Referenced Properties/Methods for SNIA_SBProtocolEndpoint (Initiator)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	See <i>Information Technology – Storage Management, Part 2 Common Architecture</i> 8.6.3 Standard Formats for Port Names.
ProtocolIFType (overridden)	Mandatory	Shall be 1 (Other).
OtherTypeDescription (overridden)	Mandatory	Shall be 'SB'.
ConnectionType (added)	Mandatory	Shall be 2 (Fibre Channel).
Role (added)	Mandatory	Shall be 2 (Initiator) or 4 (Both Initiator and Target).

21.6.14 SNIA_SBProtocolEndpoint (Target)

Target or non-local ProtocolEndpoint. The class definition specializes the CIM_ProtocolEndpoint definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 219 describes class SNIA_SBProtocolEndpoint (Target).

Table 219 - SMI Referenced Properties/Methods for SNIA_SBProtocolEndpoint (Target)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType (overridden)	Mandatory	Shall be 1 (Other).
OtherTypeDescription (overridden)	Mandatory	Shall be 'SB'.
Role (added)	Mandatory	Should be set appropriately by the instrumentation. If not know, use 0 (Unknown).
ConnectionType (added)	Mandatory	Shall be 2 (Fibre Channel).

EXPERIMENTAL

DEPRECATED

22 Backend Ports Subprofile

The functionality of the Backend Ports Subprofile has been subsumed by 18 FC Initiator Ports Profile.

The Backend Ports Subprofile is defined in 7.3.3.13 of SMI-S 1.0.2. Any instrumentation that complies to the Fibre Channel Initiator Port Profile defined in this International Standard may also claim compliance to that version of the Backend Ports Subprofile and may register as both a 1.2.0 Fibre Channel Initiator Port Subprofile and 1.0.2 Backend Ports Subprofile.

DEPRECATED

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

EXPERIMENTAL

23 FCoE Initiator Ports Profile

23.1 Synopsis

Profile Name: FCoE Initiator Ports (Component Profile)

Version: 1.0.0

Organization: SNIA

CIM Schema Version: 2.9.0

Related Profiles for FCoE Initiator Ports: Not defined in this International Standard.

Specializes: Generic Initiator Ports Profile

Central Class: CIM_FCPort

Scoping Class: CIM_ComputerSystem in the Base Server Profile (or some other autonomous profile)

The FCoE Initiator Ports Profile models the behavior of the Fibre Channel over Ethernet (FCoE) functionality of a Converged Network Adaptor (CNA).

23.2 Description

The FCoE Initiator Ports Profile is a component profile that models the behavior of the Fibre Channel over Ethernet (FCoE) functionality of a Converged Network Adaptor (CNA).

A CNA may support functionality beyond FCoE, including generic TCP/IP support. Functionality of CNAs other than FCoE is outside the scope of this profile.

23.3 Implementation

23.3.1 General

Figure 29 shows the model for classes in the FCoE Initiator Ports Profile

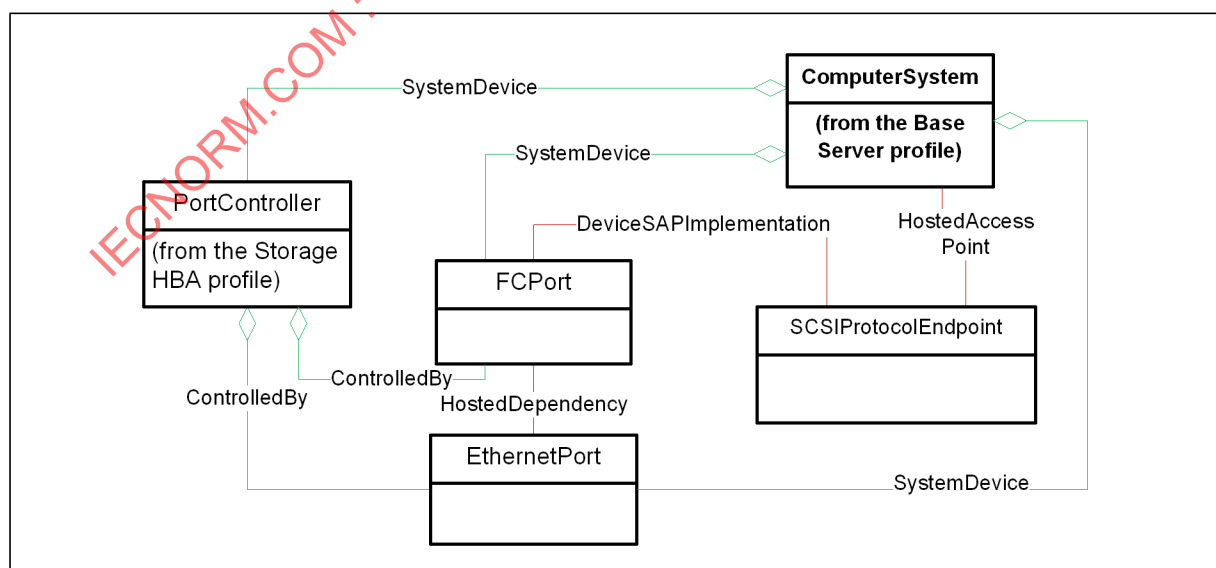


Figure 29 - FCoE Initiator Instance Diagram

Each FCoE port shall be modeled with single instance each of FCPort, EthernetPort, and SCSIProtocolEndpoint. FCPort and EthernetPort shall be associated with HostedDependency.

FCPort and EthernetPort shall each be associated to the ComputerSystem in the Base Server Profile with SystemDevice. FCPort and SCSIProtocolEndpoint shall be associated with DeviceSAPImplementation. SCSIProtocolEndpoint shall be associated to the ComputerSystem (defined in the Base Server Profile) with HostedAccessPoint.

23.3.2 Relationship to Storage HBA Profile

The FCoE Initiator ports profile is used in conjunction with the Storage HBA Profile (see *Information Technology – Storage Management, Part 7 Host Elements 7 Storage HBA Profile*). The Storage HBA Profile models the management of HBA cards independent of connectivity, and the FCoE Ethernet Ports Profile models the management of FCoE ports on an HBA.

The ControlledBy association defined in *Information Technology – Storage Management, Part 7 Host Elements 7 Storage HBA Profile* shall reference instance of FCPort and shall not reference instances of EthernetPort.

23.3.3 Optional target model

Figure 30 shows an example of a single port and drive connected to a single system using Fibre Channel. This instance diagram shows a disk (LogicalDevice in the diagram would be subclassed as something like StorageExtent) in an array, connected by a Fibre Channel port. The full model for the disk is shown in 12 Disk Drive Lite Subprofile.

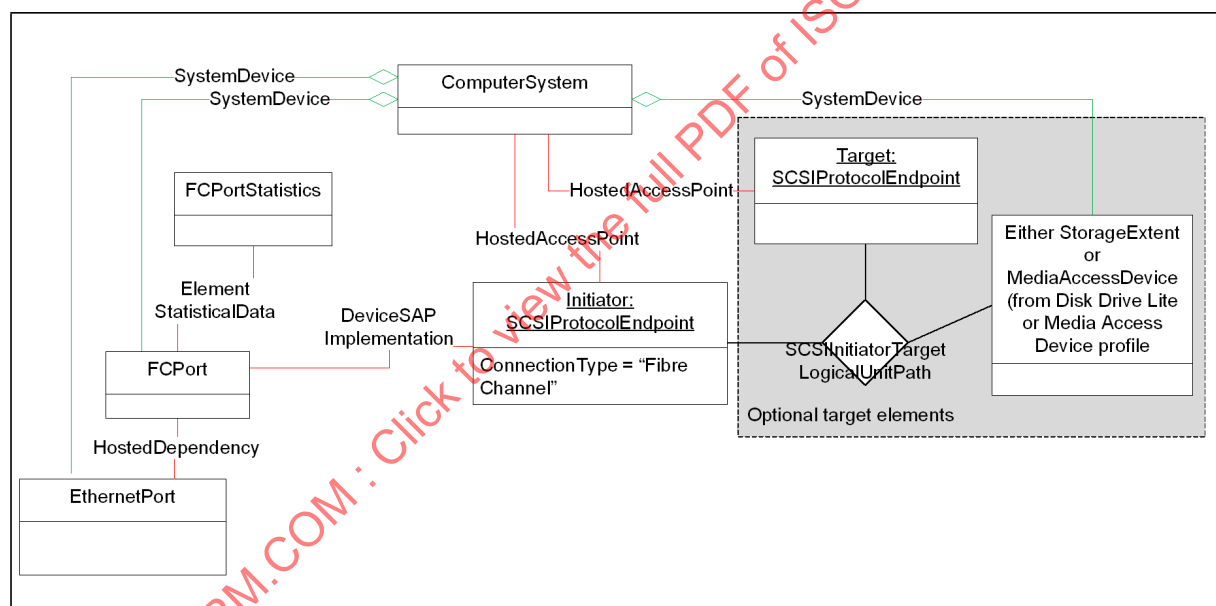


Figure 30 - Optional Target Element Model

23.3.4 Port Statistics

The FCPortStatistics subclass of NetworkPortStatistics is optional. If supported, FCPortStatistics shall be associated to FcPort using ElementStatisticalData.

23.3.5 Logical Port Group (FC Node)

LogicalPortGroup may optionally be used to model the collection of ports that shared a Node WWN (in this case, both ports on a card, but other implementations are in use). If LogicalPortGroup is instantiated, it shall be associated to the ComputerSystem in the referencing profile using HostedCollection and also associated to FCPorts using MemberOfCollection. Figure 31 shows the logical port group model for FC Nodes.

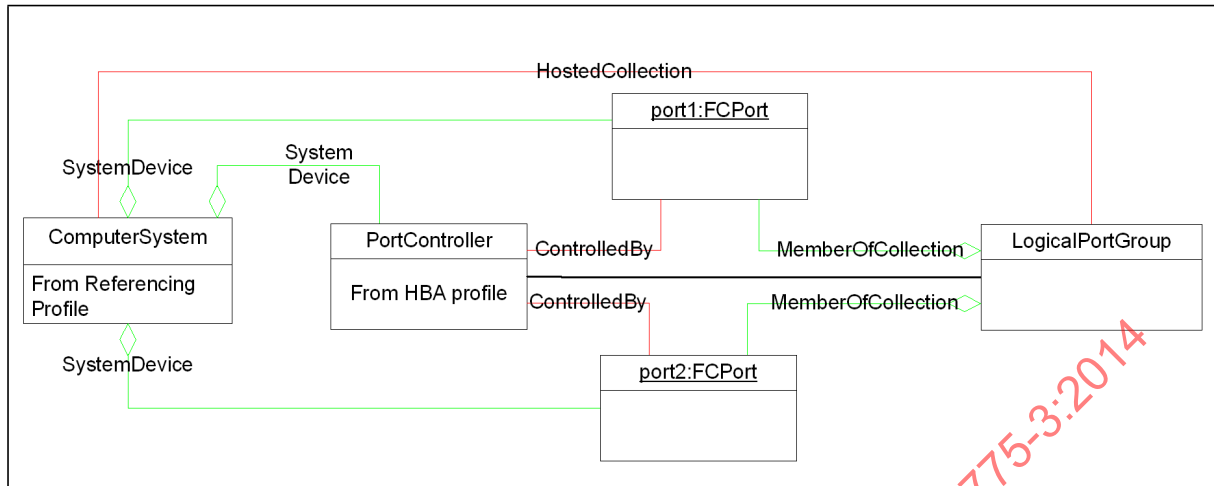


Figure 31 - Logical Port Group Model

23.3.6 Health and Fault Management Considerations

Table 220 summarized the Health and Fault Management considerations specific to this profile.

Table 220 - FCPort OperationalStatus

OperationalStatus	Description
(2) OK	Port is online
(6) Error	Port has a failure
(10) Stopped	Port is disabled
(11) InService	Port is in Self Test
(0) Unknown	

23.3.7 Cascading Considerations

Not defined in this International Standard.

23.4 Methods

23.4.1 Extrinsic Methods of this Profile

None.

23.4.2 Intrinsic Methods of this Profile

The profile supports read methods and association traversal. Specifically, the list of intrinsic operations supported are as follows:

- GetInstance
- Associators
- AssociatorNames
- References
- ReferenceNames
- EnumerateInstances

- EnumerateInstanceNames

23.5 Detailed Use Cases and Recipes

None.

23.6 CIM Elements

23.6.1 General

Table 221 describes the CIM elements for FCoE Initiator Ports.

Table 221 - CIM Elements for FCoE Initiator Ports

Element Name	Requirement	Description
23.6.2 CIM_ConnectivityCollection	Optional	Represents a collection of connected ProtocolEndpoints.
23.6.3 CIM_DeviceSAPImplementation	Mandatory	Connects Initiator LogicalPort and ProtocolEndpoint.
23.6.4 CIM_ElementStatisticalData (Port Statistics)	Optional	Connects LogicalPort and LogicalPortStatistics.
23.6.5 CIM_EthernetPort	Mandatory	
23.6.6 CIM_FCPort	Mandatory	Represents the logical aspects of the physical port and may have multiple associated protocols.
23.6.7 CIM_FCPortStatistics	Mandatory	Statistics for a port.
23.6.8 CIM_HostedAccessPoint (Initiator)	Mandatory	Associates system to initiator protocol endpoints.
23.6.9 CIM_HostedAccessPoint (Target)	Optional	Associates system to optional remote protocol endpoints.
23.6.10 CIM_HostedCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Associates the ConnectivityCollection to the hosting System.
23.6.11 CIM_HostedCollection (FC Node)	Optional	Associates the LogicalPortGroup (Fibre Channel Node) to the hosting System.
23.6.12 CIM_HostedDependency (NetworkPort to FCPort)	Mandatory	Association between EthernetPort and FCPort.
23.6.13 CIM_LogicalPortGroup	Optional	Collection of Fibre Channel ports that share a Node WWN.
23.6.14 CIM_MemberOfCollection (Connectivity Collection)	Conditional	Conditional requirement: Support for ConnectivityCollections. Associates ProtocolEndpoints to the ConnectivityCollection.
23.6.15 CIM_MemberOfCollection (FC Node)	Optional	Associates FCPort to the LogicalPortGroup.
23.6.16 CIM_ProtocolEndpoint (Initiator)	Mandatory	Represents a protocol (command set) supported by the port. The appropriate subclass (SCSIProtocolEndpoint, ATAProtocolEndpoint, SBProtocolEndpoint) should be used in initiator port specialized profiles.
23.6.17 CIM_ProtocolEndpoint (Target)	Optional	Models protocols of remote ports - target devices and possibly other initiators.
23.6.18 CIM_SCSIInitiatorTargetLogicalUnitPath	Optional	Represents a path between a SCSI initiator, target, and logical unit.
23.6.19 CIM_SCSIProtocolEndpoint (Initiator)	Mandatory	

Table 221 - CIM Elements for FCoE Initiator Ports

Element Name	Requirement	Description
23.6.20 CIM_SCSIProtocolEndpoint (Target)	Optional	Models remote ports - target devices and possibly other initiators.
23.6.21 CIM_SystemDevice (Ethernet Port)	Mandatory	Associates system to ethernet ports.
23.6.22 CIM_SystemDevice (Initiator Ports)	Mandatory	Associates system to FCPorts.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_FCPort	Optional	CQL -Creation of an FCoE Port.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.CIM_FCPort::OperationalStatus <> PreviousInstance.CIM_FCPort::OperationalStatus	Optional	CQL -Modify FCPort.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_FCPort	Optional	CQL -Deletion of an FCoE Port.

23.6.2 CIM_ConnectivityCollection

Represents a collection of connected ProtocolEndpoints.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 222 describes class CIM_ConnectivityCollection.

Table 222 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection

Properties	Requirement	Description and Notes
InstanceID	Mandatory	

23.6.3 CIM_DeviceSAPImplementation

Connects Initiator LogicalPort and ProtocolEndpoint. The class definition specializes the CIM_DeviceSAPImplementation definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 223 describes class CIM_DeviceSAPImplementation.

Table 223 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Requirement	Description and Notes
Dependent (overridden)	Mandatory	Reference to SCSIProtocolEndpoint.
Antecedent (overridden)	Mandatory	Reference to FCPort.

23.6.4 CIM_ElementStatisticalData (Port Statistics)

Connects LogicalPort and LogicalPortStatistics. The class definition specializes the CIM_ElementStatisticalData definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 224 describes class CIM_ElementStatisticalData (Port Statistics).

Table 224 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (Port Statistics)

Properties	Requirement	Description and Notes
ManagedElement (overridden)	Mandatory	Reference to FC subclass.
Stats (overridden)	Mandatory	Reference to FCPortStatistics.

23.6.5 CIM_EthernetPort

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 225 describes class CIM_EthernetPort.

Table 225 - SMI Referenced Properties/Methods for CIM_EthernetPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
LinkTechnology	Mandatory	Shall be 2 (Ethernet).
OperationalStatus	Mandatory	Shall be 0 (Unknown), 2 (OK), 6 (Error), 10 (Stopped), or 11 (In Service).
PermanentAddress	Mandatory	The MAC Address. Shall be formatted as 12 un-separated upper case hex digits.

23.6.6 CIM_FCPort

Represents the logical aspects of the physical port and may have multiple associated protocols. The class definition specializes the CIM_LogicalPort definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 226 describes class CIM_FCPort.

Table 226 - SMI Referenced Properties/Methods for CIM_FCPort

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
OperationalStatus	Mandatory	
UsageRestriction	Mandatory	Shall be 3 for ports restricted to back-end (initiator) only or 4 if the port is unrestricted.
PortType (overridden)	Mandatory	Shall be 0 (Unknown), 1 (Other), N (10), or Nx (13).
ElementName (added)	Mandatory	Port Symbolic Name.
Speed (added)	Mandatory	Speed in bits per second. Shall be 0, 1062500000 (1GFC), 2125000000 (2GFC), 4250000000 (4GFC), 8500000000 (8GFC), 10312500000 (10GE/10GFCoE), 14025000000 (16GFC), or 28500000000 (32GFC).
MaxSpeed (added)	Mandatory	Maximum Port Speed.
PortNumber (added)	Optional	
PermanentAddress (added)	Optional	Port WWN. PermanentAddress is optional when used as a back-end port in a device. This may be overridden in profiles that use this profile. Shall be 16 un-separated upper case hex digits.
NetworkAddresses (added)	Optional	For Fibre Channel end device ports, the Fibre Channel ID. Shall be 16 un-separated upper case hex digits.
SupportedCOS (added)	Optional	List of supported classes of services. Shall include 0 (unknown), 1 (Class 1), 2 (Class 2), 3, (Class 3), 4 (Class 4), 6 (Class 6), or 7 (Class 7).
ActiveCOS (added)	Optional	List of active classes of services. Shall include 0 (unknown), 1 (Class 1), 2 (Class 2), 3, (Class 3), 4 (Class 4), 6 (Class 6), or 7 (Class 7).
SupportedFC4Types (added)	Optional	
ActiveFC4Types (added)	Optional	
LinkTechnology (added)	Mandatory	Shall be 4 (FC).
SupportedMaximumTransmissionUnit (added)	Mandatory	
ActiveMaximumTransmissionUnit (added)	Optional	

23.6.7 CIM_FCPortStatistics

The class definition specializes the SNIA_LogicalPortStatistics definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 227 describes class CIM_FCPortStatistics.

Table 227 - SMI Referenced Properties/Methods for CIM_FCPortStatistics

Properties	Requirement	Description and Notes
ElementName	Mandatory	
InstanceID	Mandatory	
BytesTransmitted	Mandatory	
BytesReceived	Mandatory	
PacketsTransmitted	Mandatory	
PacketsReceived	Mandatory	
CRCErrors (added)	Mandatory	Maps to HBA API HBA_PortStatistics.InvalidCRCCCount.
LinkFailures (added)	Mandatory	Maps to HBA API HBA_PortStatistics.LinkFailureCount.
PrimitiveSeqProtocolErrCount (added)	Mandatory	
LossOfSignalCounter (added)	Mandatory	Maps to HBA API HBA_PortStatistics.LossOfSignalCount.
InvalidTransmissionWords (added)	Mandatory	Maps to HBA API HBA_PortStatistics.InvalidTxWordCount.
StatisticTime (added)	Optional	Time last measurement was taken.
LIPCount (added)	Mandatory	
NOSCount (added)	Mandatory	
ErrorFrames (added)	Mandatory	
DumpedFrames (added)	Mandatory	
LossOfSyncCounter (added)	Mandatory	Maps to HBA API HBA_PortStatistics.LossOfSynchCount.

23.6.8 CIM_HostedAccessPoint (Initiator)

Associates system to initiator protocol endpoints.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 228 describes class CIM_HostedAccessPoint (Initiator).

Table 228 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Initiator)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent	Mandatory	Reference to ProtocolEndpoint(Initiator).

23.6.9 CIM_HostedAccessPoint (Target)

Associates system to optional remote protocol endpoints.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 229 describes class CIM_HostedAccessPoint (Target).

Table 229 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (Target)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.
Dependent	Mandatory	Reference to ProtocolEndpoint(Target).

23.6.10 CIM_HostedCollection (Connectivity Collection)

Associates the ConnectivityCollection to the hosting System.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 230 describes class CIM_HostedCollection (Connectivity Collection).

Table 230 - SMI Referenced Properties/Methods for CIM_HostedCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Dependent	Mandatory	Reference to ConnectivityCollection.
Antecedent	Mandatory	Reference to ComputerSystem in referencing profile.

23.6.11 CIM_HostedCollection (FC Node)

Associates the LogicalPortGroup (Fibre Channel Node) to the hosting System.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 231 describes class CIM_HostedCollection (FC Node).

Table 231 - SMI Referenced Properties/Methods for CIM_HostedCollection (FC Node)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to ComputerSystem.
Dependent	Mandatory	Reference to LogicalPortGroup.

23.6.12 CIM_HostedDependency (NetworkPort to FCPort)

Association between EthernetPort and FCPort.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 232 describes class CIM_HostedDependency (NetworkPort to FCPort).

Table 232 - SMI Referenced Properties/Methods for CIM_HostedDependency (NetworkPort to FCPort)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to EthernetPort.
Dependent	Mandatory	Reference to FCPort.

23.6.13 CIM_LogicalPortGroup

Represents the Fibre Channel Node

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 233 describes class CIM_LogicalPortGroup.

Table 233 - SMI Referenced Properties/Methods for CIM_LogicalPortGroup

Properties	Requirement	Description and Notes
InstanceID	Mandatory	Opaque.
Name	Mandatory	Fibre Channel Node WWN.
NameFormat	Mandatory	Shall be 'WWN'.
ElementName	Mandatory	Node Symbolic Name.

23.6.14 CIM_MemberOfCollection (Connectivity Collection)

Associates ProtocolEndpoints to the ConnectivityCollection. The class definition specializes the CIM_MemberOfCollection definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ConnectivityCollections.

Table 234 describes class CIM_MemberOfCollection (Connectivity Collection).

Table 234 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Connectivity Collection)

Properties	Requirement	Description and Notes
Member (overridden)	Mandatory	Reference to ProtocolEndpoint.
Collection	Mandatory	Reference to ConnectivityCollection.

23.6.15 CIM_MemberOfCollection (FC Node)

Associates FCPort to the LogicalPortGroup.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 235 describes class CIM_MemberOfCollection (FC Node).

Table 235 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (FC Node)

Properties	Requirement	Description and Notes
Collection	Mandatory	Reference to LogicalPortGroup.
Member	Mandatory	Reference to FCPort.

23.6.16 CIM_ProtocolEndpoint (Initiator)

Represents a protocol (command set) supported by the port.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 236 describes class CIM_ProtocolEndpoint (Initiator).

Table 236 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint (Initiator)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	

Table 236 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint (Initiator)

Properties	Requirement	Description and Notes
Name	Mandatory	See <i>Information Technology – Storage Management, Part 2 Common Architecture</i> 8.6.3 Standard Formats for Port Names.
ProtocolIFType	Mandatory	Shall be 1 (Other).
OtherTypeDescription	Mandatory	Shall be the string 'SCSI', 'ATA', or 'SB'. Initiator port specialized profiles specify the appropriate subset.

23.6.17 CIM_ProtocolEndpoint (Target)

Models protocols of remote ports - target devices and possibly other initiators. The appropriate subclass (SCSIProtocolEndpoint, ATAProtocolEndpoint, SBProtocolEndpoint) should be used in initiator port specialized profiles.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 237 describes class CIM_ProtocolEndpoint (Target).

Table 237 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint (Target)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType	Mandatory	The values in MOFs map to IETF values and exclude storage. Shall be 1 (Other) and set OtherTypeDescription appropriately.
OtherTypeDescription	Mandatory	Shall be the string 'SCSI', 'ATA', or 'SB'. Initiator port specialized profiles specify the appropriate subset.

23.6.18 CIM_SCSIInitiatorTargetLogicalUnitPath

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 238 describes class CIM_SCSIInitiatorTargetLogicalUnitPath.

Table 238 - SMI Referenced Properties/Methods for CIM_SCSIInitiatorTargetLogicalUnitPath

Properties	Requirement	Description and Notes
LogicalUnit	Mandatory	
Initiator	Mandatory	
Target	Mandatory	

23.6.19 CIM_SCSIProtocolEndpoint (Initiator)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 239 describes class CIM_SCSIProtocolEndpoint (Initiator).

Table 239 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Initiator)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
ProtocolIFType	Mandatory	Shall be 1 (Other).
OtherTypeDescription	Mandatory	Shall be the string 'SCSI'.
ConnectionType	Mandatory	Shall be 2 (Fibre Channel).
Role	Mandatory	Shall be 2 (Initiator).

23.6.20 CIM_SCSIProtocolEndpoint (Target)

Models remote ports - target devices and possibly other initiators.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 240 describes class CIM_SCSIProtocolEndpoint (Target).

Table 240 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint (Target)

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
Role	Mandatory	Should be set appropriately by the instrumentation. If not know, use 0 (Unknown).
ProtocolIFType	Mandatory	The values in MOFs map to IETF values and exclude storage. Shall be 1 (Other) and set OtherTypeDescription to 'SCSI'.
OtherTypeDescription	Mandatory	Shall be the string 'SCSI'.
ConnectionType	Mandatory	Shall be 8 (FC).

23.6.21 CIM_SystemDevice (Ethernet Port)

Associates system to ethernet ports.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 241 describes class CIM_SystemDevice (Ethernet Port).

Table 241 - SMI Referenced Properties/Methods for CIM_SystemDevice (Ethernet Port)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to ComputerSystem.
PartComponent	Mandatory	Reference to EthernetPort.

23.6.22 CIM_SystemDevice (Initiator Ports)

Associates system to FCPorts. The class definition specializes the CIM_SystemDevice definition in the Generic Initiator Ports profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 242 describes class CIM_SystemDevice (Initiator Ports).

Table 242 - SMI Referenced Properties/Methods for CIM_SystemDevice (Initiator Ports)

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to ComputerSystem.
PartComponent (overridden)	Mandatory	Reference to FCPort.

EXPERIMENTAL

STABLE

24 Access Points Subprofile

24.1 Description

The Access Points Subprofile provides addresses of remote access points for management services.

This is modeled using a RemoteServiceAccessPoint linked to the managed system using a HostedAccessPoint association.

A management service is typically associated with all elements in a system, but in some cases, a management service relates to a subset of elements. The scope of a RemoteServiceAccessPoint may be constrained to a subset of elements using SAPAvailableForElement. If the service referenced in RemoteServiceAccessPoint is not referenced by any SAPAvailableForElement associations, then the service described by RemoteServiceAccessPoint shall apply to all the elements of the system referenced via HostedAccessPoints. This type of system-wide service is depicted in Figure 32.

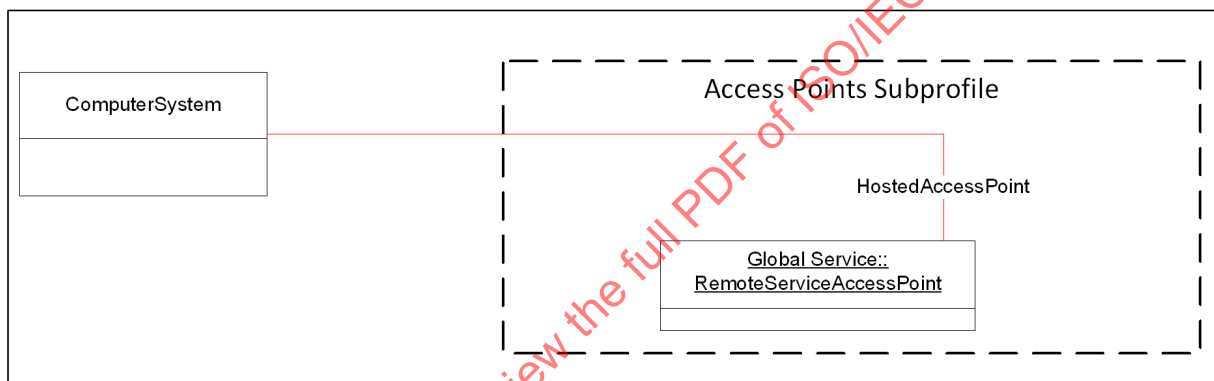


Figure 32 - System-wide Remote Access Point

If the service referenced in RemoteServiceAccessPoint is referenced by any SAPAvailableForElement associations, then the service described by RemoteServiceAccessPoint shall apply to the subset of elements referenced via SAPAvailableForElement associations. The HostedAccessPoint association between RemoteServiceAccessPoint is still mandatory (so the client can readily associate the service to a specific storage system).

Figure 33 depicts a configuration with two RemoteServiceAccessPoint instances. One represents a system-wide service and the other represents a service that applies just to certain devices.

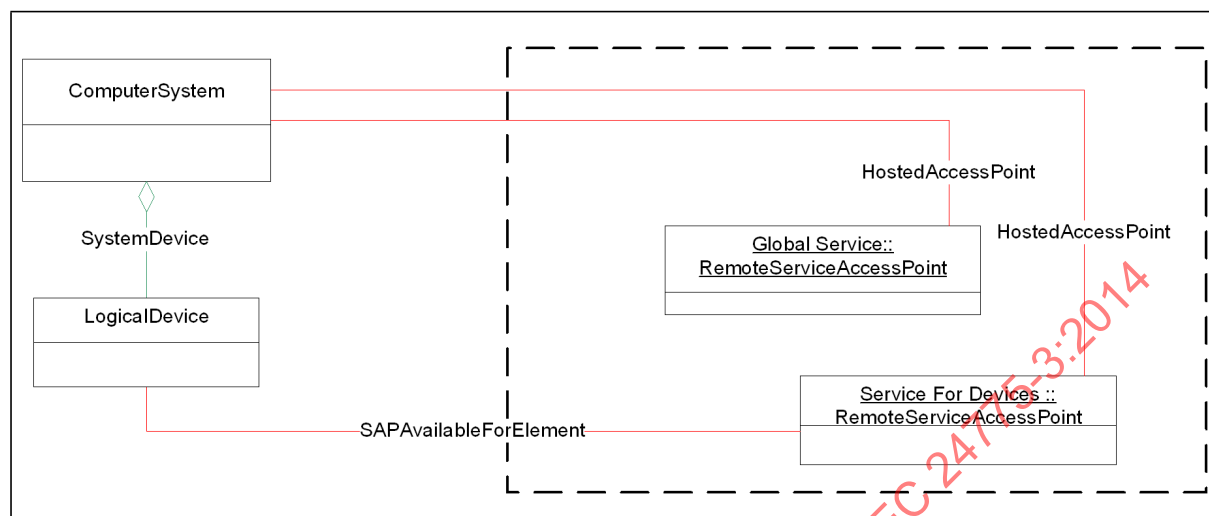


Figure 33 - Access Point Instance Diagram

The exposed management services may represent a web UI that can be launched by a web browser, a telnet interface, or some vendor-specific interface. RemoteServiceAccessPoint InfoFormat property describes the format of the AccessInfo property; valid options include "URL" and "FQDN". In a URL, the text before the "://" is referred to as the "scheme". A URL with an http or HTTPS scheme is often a web/HTML page, but HTTP can be used for other purposes. Table 243 specifies the requirements for InfoFormat, AccessInfo, and the scheme subset of a URL AccessInfo.

Table 243 - RemoteAccessPoint InfoFormat and AccessInfo Properties

InfoFormat	AccessInfo Scheme	Description
"URL"	"http" or "https"	The references URL shall be a valid web page. It should provide element management for the system or elements referenced by the associated HostedAccessPoint association.
"Other" with OtherInfoFormatDescription = "Non-UI URL"	"http" or "https"	Used for HTTP URLs that do not reference a valid web UI.
"URL"	anything other than "http" and "https"	May be used. No standard behavior is specified.
others from the MOF	n/a	May be used. No standard behavior is specified.

24.2 Health and Fault Management Considerations

Not defined in this International Standard.

24.3 Cascading Considerations

Not defined in this International Standard.

24.4 Supported Subprofiles and Packages

Not defined in this International Standard.

24.5 Methods of this Profile

Not defined in this International Standard.

24.6 Client Considerations and Recipes

Not defined in this International Standard.

24.7 Registered Name and Version

Access Points version 1.3.0 (Component Profile)

24.8 CIM Elements

24.8.1 General

Table 244 describes the CIM elements for Access Points.

Table 244 - CIM Elements for Access Points

Element Name	Requirement	Description
24.8.2 CIM_HostedAccessPoint	Mandatory	Associate the RemoteServiceAccessPoint to the System on which it is hosted.
24.8.3 CIM_RemoteServiceAccessPoint	Mandatory	A ServiceAccessPoint for management tools.
24.8.4 CIM_SAPAvailableForElement	Optional	This association identifies the element that is serviced by the RemoteServiceAccessPoint.

24.8.2 CIM_HostedAccessPoint

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 245 describes class CIM_HostedAccessPoint.

Table 245 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint

Properties	Flags	Requirement	Description and Notes
Antecedent		Mandatory	The Hosting System.
Dependent		Mandatory	The access point(s) that are hosted on this System.

24.8.3 CIM_RemoteServiceAccessPoint

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 246 describes class CIM_RemoteServiceAccessPoint.

Table 246 - SMI Referenced Properties/Methods for CIM_RemoteServiceAccessPoint

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
Name		Mandatory	
ElementName		Mandatory	User Friendly name.
AccessInfo		Mandatory	Management Address.
InfoFormat		Mandatory	The format of the Management Address. For interoperability, this shall be 'URL' (200).

24.8.4 CIM_SAPAvailableForElement

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 247 describes class CIM_SAPAvailableForElement.

Table 247 - SMI Referenced Properties/Methods for CIM_SAPAvailableForElement

Properties	Flags	Requirement	Description and Notes
ManagedElement		Mandatory	The managed element.
AvailableSAP		Mandatory	The service access point.

STABLE

DEPRECATED

25 Cascading Subprofile

25.1 Introduction to cascading subprofile

The Cascading Subprofile is scheduled for removal for SMI-S 2.0. The functionality of this profile will not be replaced in SMI-S 2.0. In SMI-S 2.0, cascading elements are to be done as profile specific cascading component profile or the cascading elements are to simply be embedded in the profile that is doing the cascading.

25.2 Description

25.2.1 Overview

The Cascading Subprofile defines the set of classes, methods and behavior used to model cross profile dependencies and references. This includes modeling of cross CIM server references when the referenced profile is managed by another CIM server.

Examples of SMI-S Profiles that should support the Cascading Subprofile include Storage Virtualizer, NAS Heads and Volume Managers. However, other profiles may also support the Cascading Subprofile for cross profile references. For example, an Array Profile may support the Cascading Subprofile to effect cross profile references used in “remote copy.”

For ease of documentation, a profile that supports the Cascading Subprofile is referred to as a **cascading profile**. The profile referenced is referred to as a **leaf profile**. For example, storage virtualization would support the Cascading Subprofile and would be a cascading profile. It would reference storage volumes in one or more Array profiles. In such configurations, the Array profiles would be referred to as Leaf profiles.

The cascading subprofile defines a common approach to “stitching” resources in the cascading profile to resources in the leaf profiles. While the general mechanism used is common, the specifics may vary depending on the resources that are stitched together. For example, a Storage Virtualization Profile would stitch StorageExtents (in the virtualizer) to StorageVolumes (in arrays). But a Volume Manager would stitch LogicalDisks (in the volume manager) to StorageVolumes (in arrays or virtualizers).

The cascading subprofile defines how to model the relationships between CIM Servers when there are CIM Servers of Leaf profiles that are referenced by a CIM Server of the cascading profile, and how a client manages the interaction between CIM Servers in a cascading configuration (including CIM Server credentials).

In addition to the Cascading Subprofile, there are two related subprofiles that may also be supported by the cascading profile or the leaf profiles. They are the Credential Management Subprofile, which defines the classes, methods and behavior for managing the credentials used by a CIM server of the cascading profile when accessing (different) CIM Servers of Leaf profiles. The second is the Security Resource Ownership Subprofile (or a specialization of this subprofile) which defines the classes, methods and behavior of recording ownership in the leaf profiles. The usage of these subprofiles will be referenced in this subprofile, but their definition is contained in separate subprofile specifications.

The Cascading Subprofile provides block-level configuration management in the current version of SMI-S.

The Cascading Subprofile defines cascading of resources at the block level. That is, a Cascading Profile uses Block storage resources of the leaf profiles. These are StorageVolumes or LogicalDisks. In the current version of SMI-S the model will only be tested in the context of cascading for block storage.

25.2.2 Instance Diagrams

25.2.2.1 General

There are three aspects of the cascading subprofile that are illustrated separately:

- Logical Topology (usage of leaf resources by cascading profiles)
- Resource Allocation/Deallocation
- CIM Server Topology (usage of CIM Servers by other CIM Servers)

In addition, there are the relationships between the Cascading Subprofile and the Security Resource Ownership Subprofile and the Credential Management Subprofile. This relationship will be illustrated, but the details of those subprofiles are documented in their own sections.

25.2.2.2 Logical Topology

Figure 34 illustrates the basic constructs for modeling the logical topology represented by cascading profiles. The cascading profile is the top box. The modeling for the cascading subprofile is in the dashed box (in the Cascading Profile). The leaf profile is the lower box. Note that for the basic modeling of the logical topology of cascading, there are no modeling requirements on the leaf profile.

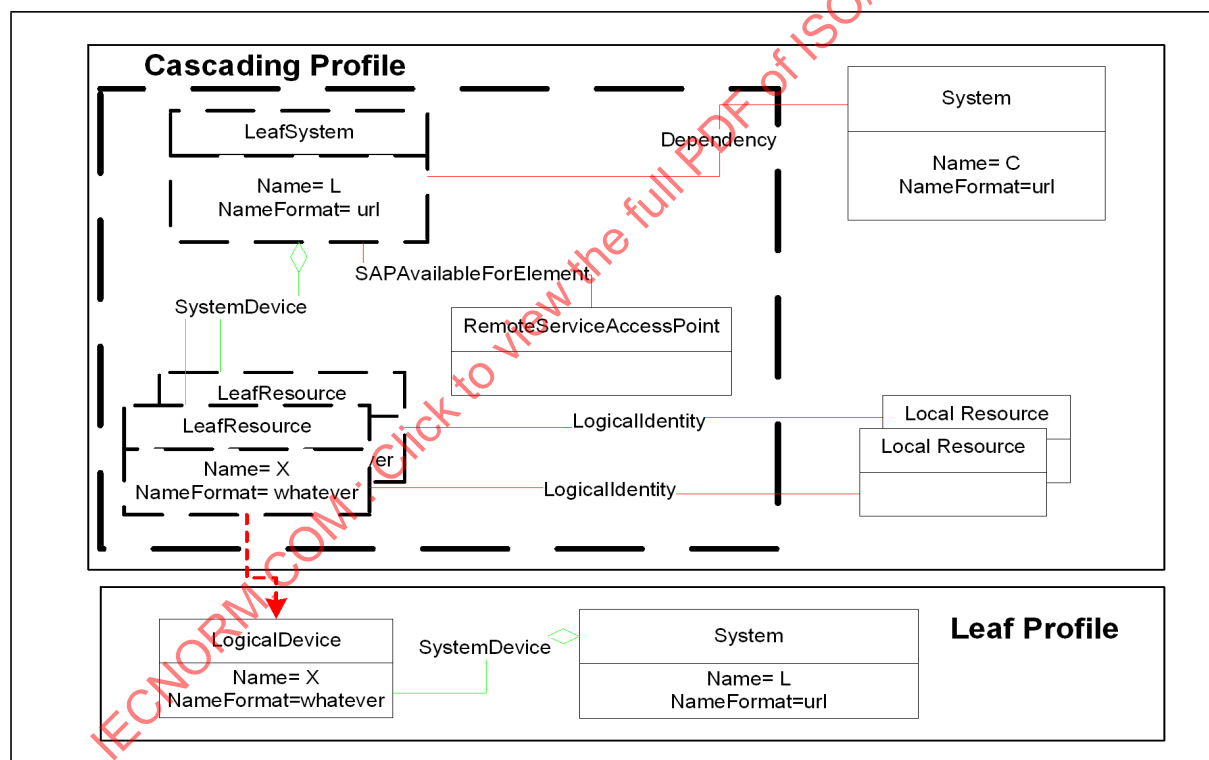


Figure 34 - Instance Diagram for Logical Topology

NOTE The dashed classes in Figure 34 are instances that are cached in the Cascading Profile. They are redundant with the instances maintained by the Leaf profile. The dashed arrows between the Cascading Profile and the Leaf Profile signifies “stitching” based on durable names or correlatable ids for the resources represented. The dashed arrows **are not** instantiated associations.

If the Cascading Subprofile is supported by the Cascading Profile, then there will be support for instantiating “leaf” “top level object” (e.g., ComputerSystems) and “leaf” LogicalDevices (e.g., StorageVolumes) in those Leaf Profiles that are “visible” to the Cascading Profile (device). The instances of the “leaf” “top level object” can be found by traversing the CascadingDependency association from the “top level object” of the Cascading Profile.

The leaf resources (logical devices) that are visible to the Cascading Profile have an association (e.g., SystemDevice association) to the “leaf” top level object (e.g., ComputerSystem) that has exposed them to the Cascading Profile.

The top level object, Hosted or SystemDevice association and LogicalDevices mirrors information that is in the Leaf Profile. In some Cascading Profile configurations, the Cascading Profile may want to subscribe to life cycle indications on the devices of interest in the Leaf Profile. However, that is a consideration of the Cascading Profile. It is not required as part of the Cascading Subprofile.

From the top level object (e.g., ComputerSystem) of the Leaf, there may be a SAPAvailableForElement association to a RemoteServiceAccessPoint instance. The RemoteServiceAccessPoint identifies information need for access to the management interface to the Leaf system. This management interface may or may not be a CIM interface.

The expectation is that the model represented in Figure 34 will be automatically maintained by the Cascading Profile (and providers). There are no methods for client manipulation of this model. In the case of the RemoteServiceAccessPoint instance, the expectation is that discovery of leaf systems would be an automatic process (e.g., SLP discovery of SMI-S Profiles and Servers) and that the provider would record the access information based on its discovery processes.

In the simplest form of cascading, this is sufficient to model the logical topology of the cascading. However, many implementations will need to go further (see 25.2.2.3).

25.2.2.3 Resource Allocation/Deallocation

In some cascading environments, it is necessary to distinguish between resources that are “visible” to the Cascading Profile from resources that are actually “in use.” For example, a Volume Manager or storage virtualization system may be able to “see” a number of storage volumes (logical units) through its ports. But this does not necessarily mean that it has allocated and is using them. A separate step is required to “prepare” the resources for use. In the case of storage virtualization systems, this step would include assigning the storage to a storage pool in the virtualizer.

To readily discern which storage volumes (logical devices) are “visible” and which volumes are assigned, two collections are defined. The collection of “visible” resources is the “RemoteResources” collection. The collection of assigned resources is the “AllocatedResources” collection. This is illustrated in Figure 35.

The SNIA_AllocationService may or may not exist. The actual function of Allocation may be implemented as a side effect of other methods. For example, allocating a Leaf StorageVolume may occur as a side effect of CreateOrModifyStoragePool, where an extent (e.g., leaf StorageVolume) is added to a StoragePool. The semantics of CreateOrModifyStoragePool constructs all the necessary associations for the StorageExtent (and may also have the semantics of an implied allocation of the StorageVolume).

To determine if allocation or deallocation are explicit (via allocate/deallocate method calls) or implicit (side effect of another method), the client should inspect the “AsynchronousMethodsSupported” and “SynchronousMethodsSupported” properties of the SNIA_CascadingCapabilities instance for the System.

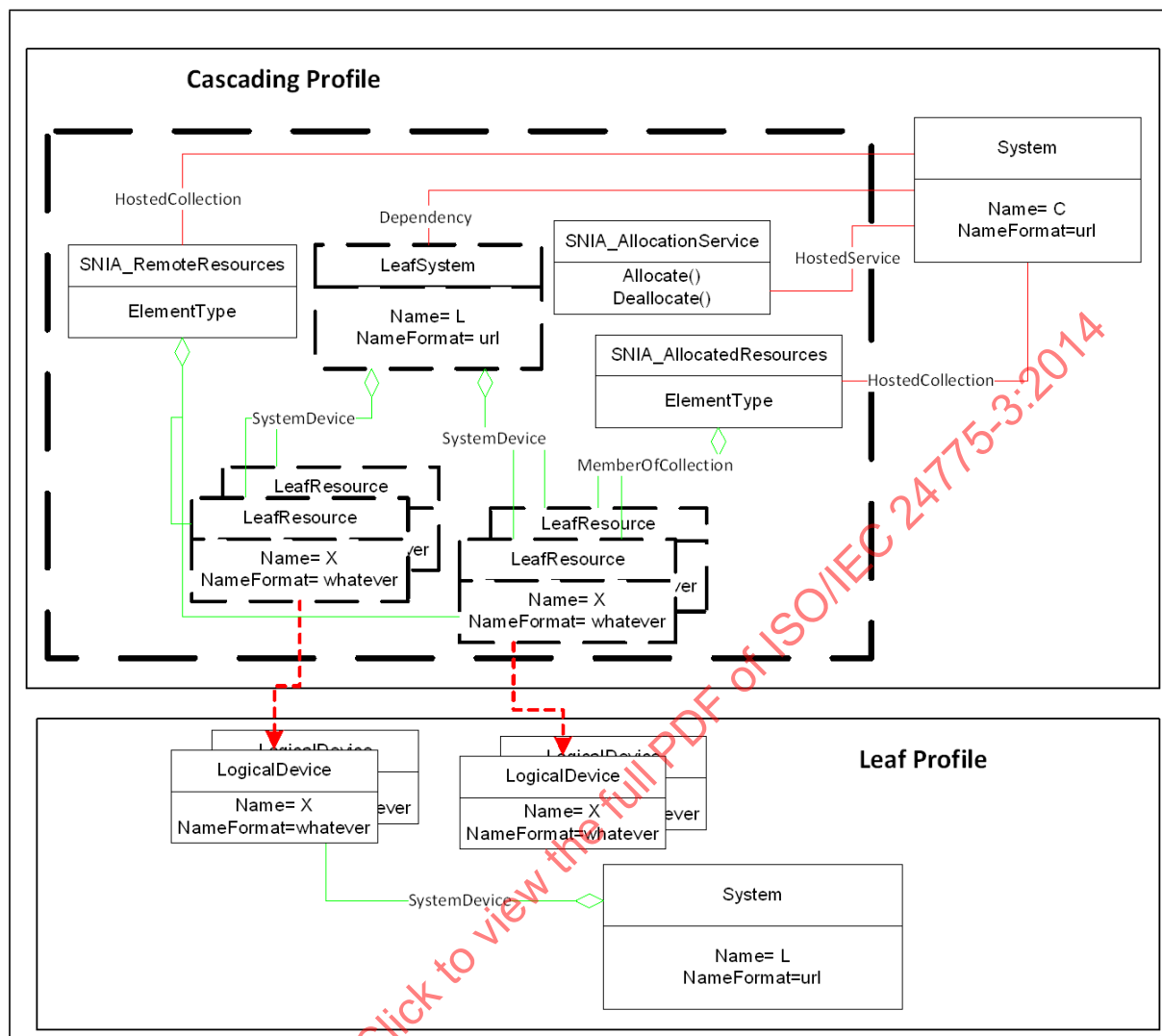


Figure 35 - Resource Allocation/Deallocation Instance Diagram

25.2.2.4 CIM Server Topology

In addition to a cascading system using leaf systems and its resources, a cascading profile may also model the dependencies between the CIM Server of the cascading profile and the CIM Servers of the Leaf Profiles. This is illustrated in Figure 36.

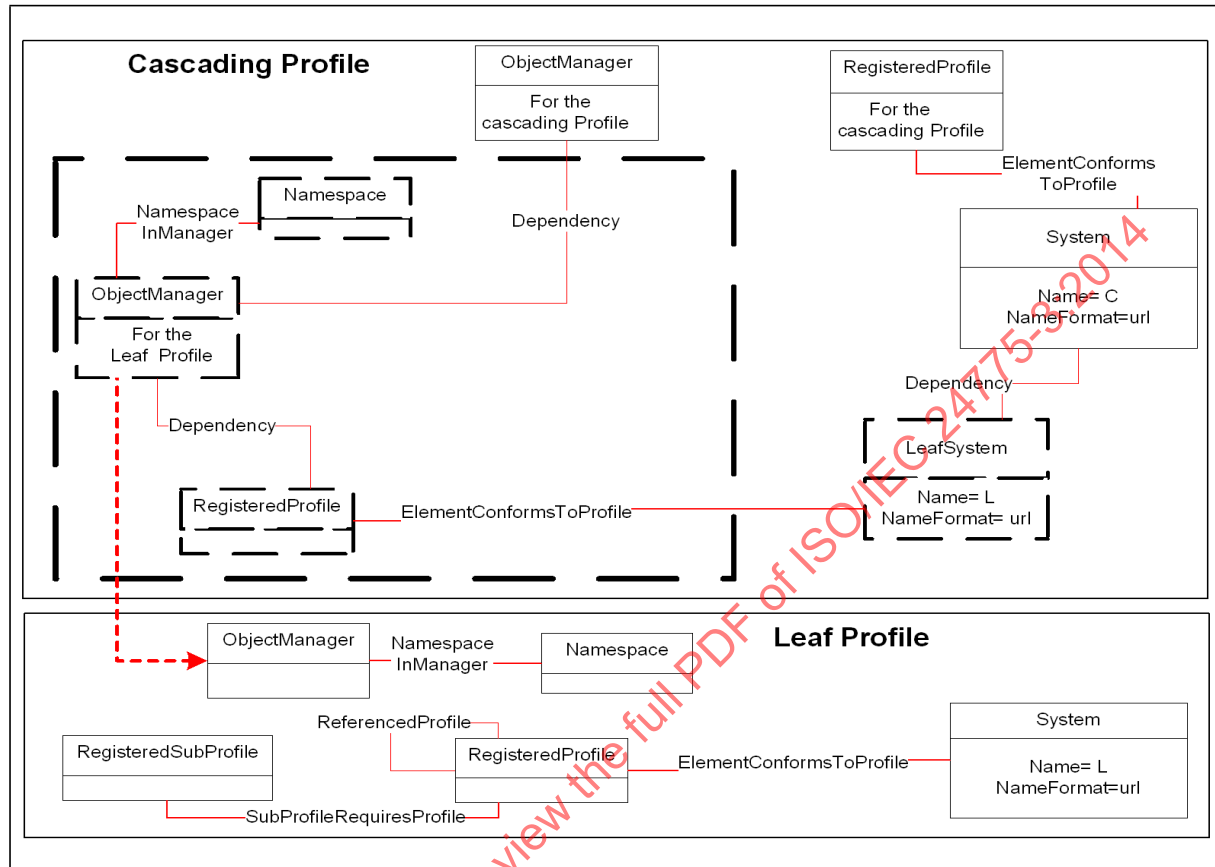


Figure 36 - Cascading Server Topology

As with the logical topology, the server topology is effected by caching Leaf information in the cascading profile. Specifically, the cached instances from the leaf profiles are:

ObjectManager – to allow the dependency between ObjectManagers to be instantiated in the cascading profile.

Namespace – to provide cached information on the namespace of the leaf CIM Server. This would be the Interop Namespace for accessing the Server Profile of the CIM Server.

RegisteredProfile – to identify the Profile of the Leaf Profile (e.g., Array or Virtualizer).

In addition, the necessary associations (HostedProfile, NamespaceInManager and ElementConformsToProfile) would be instantiated to connect the relevant instances.

The actual dependence between the CIM Server (ObjectManager) of the Cascading Profile and the CIM Server (ObjectManager) of the Leaf systems is represented by instances of Dependency.

25.2.2.5 Cascading with the Resource Ownership Subprofile

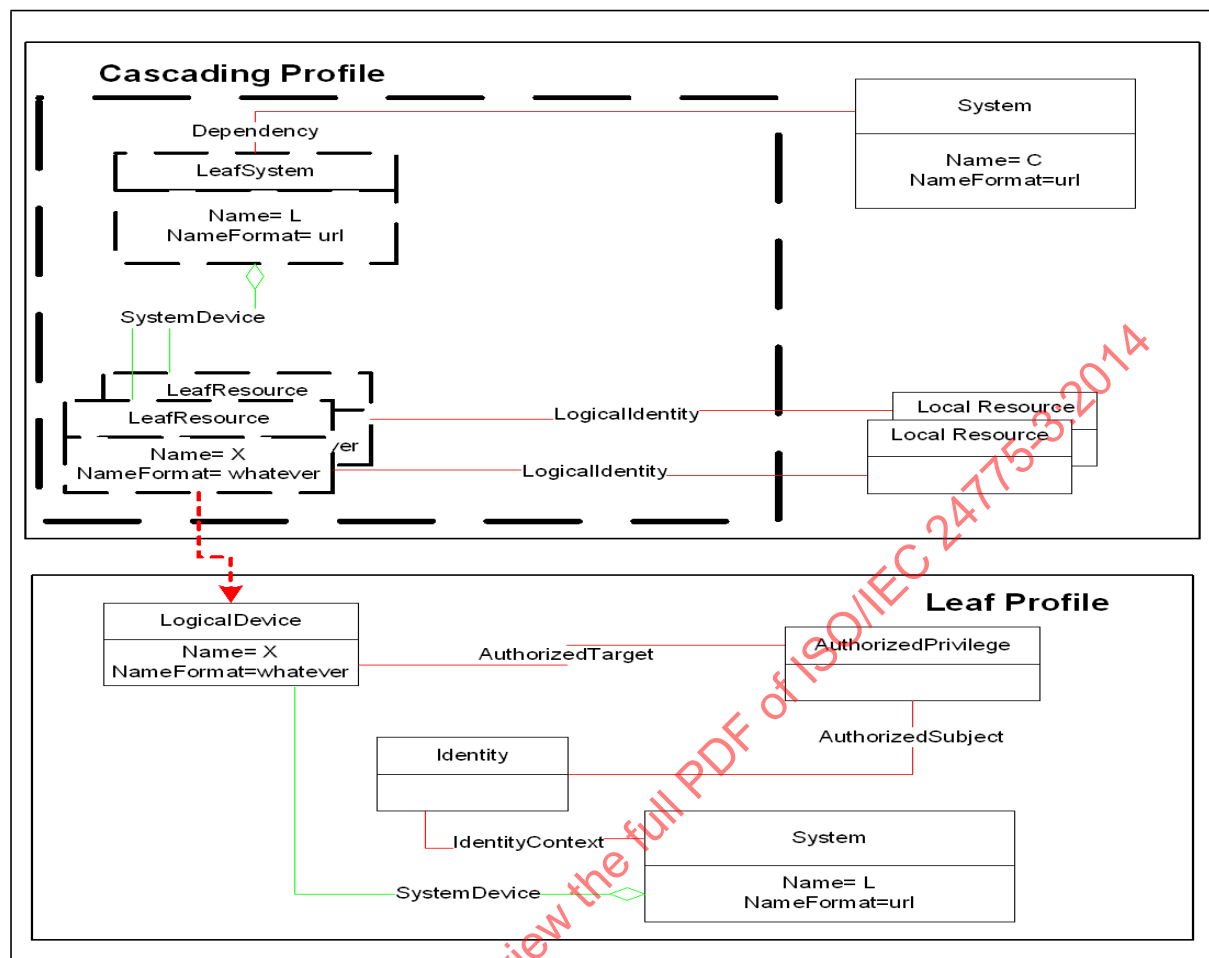


Figure 37 - Instance Diagram for Cascading with Resource Ownership

Figure 37 illustrates cascading when used in conjunction with the Security Resource Ownership Profile. The Security Resource Ownership (or a specialization of it) would be implemented in the Leaf Profile.

25.2.2.6 Cascading with the Credentials Management Subprofile

As an extension of the modeling of CIM Server topology, a cascading profile may implement the Credentials Management Subprofile. When this is done it extends the modeling for the Server topology as illustrated in Figure 38.

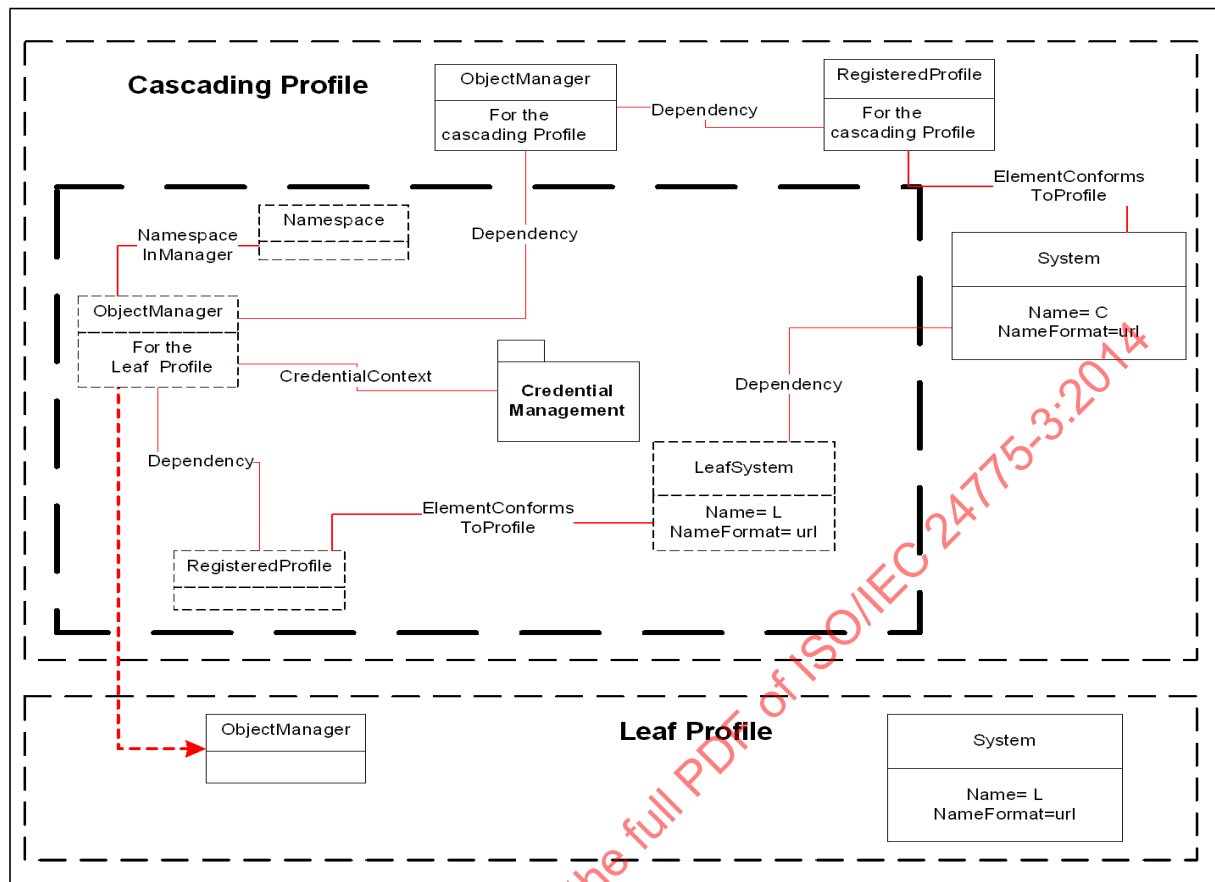


Figure 38 - Instance Diagram for Cascading with Credential Management Subprofile

The Credential Management information would be associated with the CIM Server ObjectManager instance for a Leaf system. The Credential Management Subprofile would identify how the cascading system would authenticate itself with the Leaf system.

25.2.2.7 Modeling for Defining Cascading Capabilities

As indicated in previous discussions, only parts of the Cascading subProfile are mandatory. For a list of what elements are mandatory, see Table 250. In order to make it relatively easy for clients to determine what is supported, implementation of the SNIA_CascadingCapabilities class is mandatory if cascading is supported. The modeling for this class is illustrated in Figure 39.

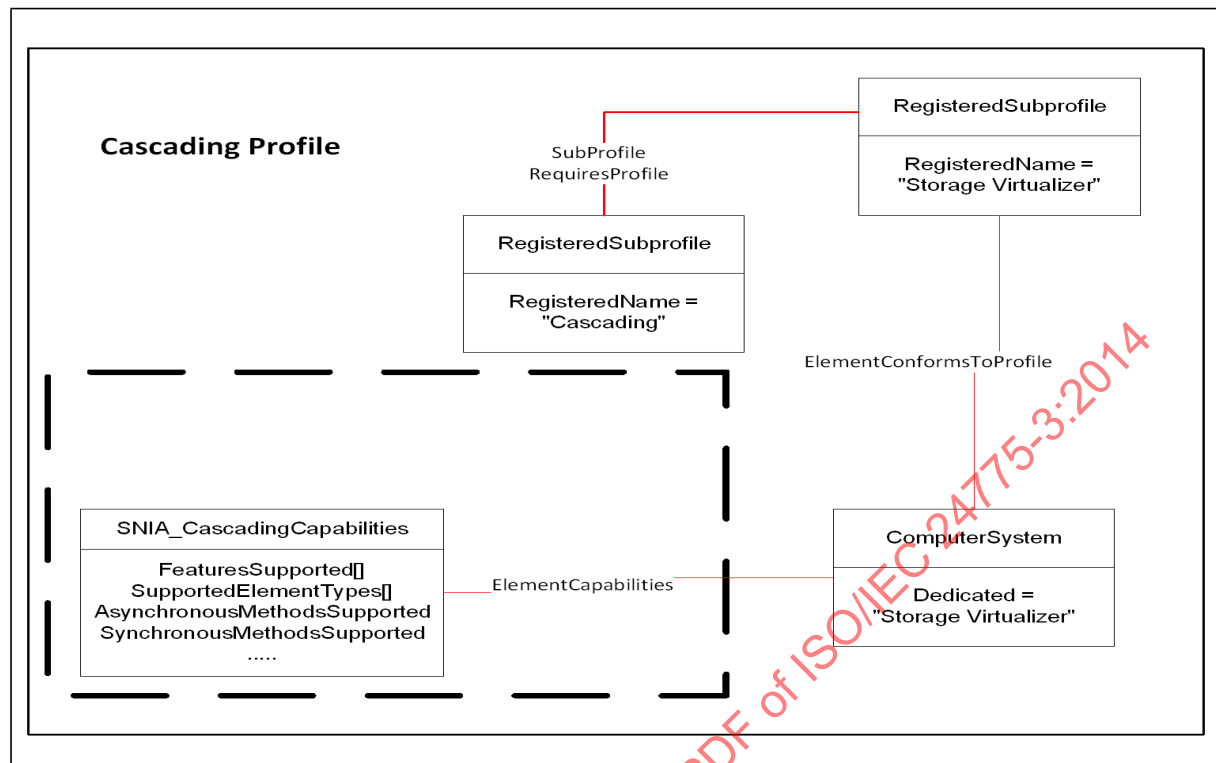


Figure 39 - Modeling of Cascading Capabilities

The SNIA_CascadingCapabilities instance would be found by doing association traversal from the RegisteredSubprofile for cascading following the ElementCapabilities association.

The properties of SNIA_CascadingCapabilities are defined as follows:

- **FeaturesSupported** - This is an array that defines the cascading features that are supported by the implementation of the Cascading Profile. The values are "Ownership", "Leaf Credentials", "OM Dependencies" and "Allocation Service".
- **SupportedElementTypes** - This is an array that defines the type of "Remote Resource" ManagedElements that are supported by the implementation. For this version of SMI-S, only StorageVolumes and LogicalDisks are supported.
- **AsynchronousMethodsSupported** – This is an array that defines any asynchronous methods supported for allocation or deallocation of leaf resources. The values are "Allocation" or "Deallocation".
- **SynchronousMethodsSupported** – This is an array that defines any synchronous methods supported for allocation or deallocation of leaf resources. The values are "Allocation" or "Deallocation".

The Cascading Subprofile uses durable names of leaf resources for stitching together the Leaf Profile and its resources to the corresponding instances in the Cascading Profile.

The CIM Server of the Cascading Profile may use indications (or provider poll on access) to keep its model accurate.

25.3 Health and Fault Management Considerations

25.3.1 Reporting Health of Leaf Systems, Resources and Object Managers

A Cascading Profile should not report health of leaf resources without verifying the health of those resources (via direct reference to the Leaf Profile). The Cascading Profile may keep

health properties in its local copy of the instances for leaf resources for its own purposes, but it should always refer to the leaf profile on requests from clients.

A request for a health property (e.g., `OperationalStatus`) should result in a request to the underlying leaf resource for the information. If the leaf resource is not available (e.g., the connection to the CIM Server is broken) the Cascading Profile may report health from its local copy of the instance.

25.3.2 Cascading Indications of Health

Given a Cascading Profile is dependent upon leaf resources, the CIM Server of the Cascading Profile may chose to subscribe to health (`OperationalStatus`) indications on the leaf resources it is actively using (allocated resources). Generally speaking, health problems on leaf resources will translate to health problems on one or more resources in the Cascading Profile. For example, if a `StorageVolume` in the Array (leaf) profile has an `OperationalStatus` of "Error", this may cause one or more `StorageVolumes` in a Virtualizer that is using the array to either be in error or be degraded.

Health indications should cascade. However, how they cascade will depend on where and how the leaf resources are used.

However a cascading profile discovers a problem with leaf resources, then it may be reflected in operational status of the cascader's resources.

25.4 Cascading Considerations

Not defined in this International Standard.

25.5 Supported Subprofiles and Packages

Table 248 describes the supported profiles for Cascading.

Table 248 - Supported Profiles for Cascading

Profile Name	Organization	Version	Requirement	Description
Server	SNIA	1.5.0	Mandatory	

25.6 Methods of this Subprofile

25.6.1 General

Table 249 summarized the extrinsic methods supported by the Cascading Subprofile.

Table 249 - Extrinsic Methods Supported by Cascading Subprofile

Method	Created Instances	Deleted Instances	Modified Instances
Allocate	MemberOfCollection	N/A	N/A
Deallocate	N/A	MemberOfCollection	N/A

25.6.2 Allocate

Starts a job to allocate remote resources (from the `RemoteResources` collection) to the `AllocatedResources` collection.

Allocate (

[IN, Description (Enumeration indicating the type of element being allocated. This type value shall match the type of the instances.),

ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8" },

Values { "Unknown", "Reserved", "Any Type", "StorageVolume", "StorageExtent", "StoragePool", "ComputerSystem", "LogicalDisk", "FileShare" }}

Only "3" (StorageVolume) is supported in this version of SMI-S.

uint16 **ElementType**;

[IN (false), OUT, Description (Reference to the job (may be null if job completed).)]

CIM_ConcreteJob REF **Job**,

[IN, Description (The reference to the AllocatedResource collection to which Elements are being added.)]

SNIA_AllocatedResources REF **Collection**,

[IN, Description (Array of strings containing representations of references to CIM_ManagedElement instances, that are being allocated to the AllocatedResources Collection.)]

string **InElements[]**;

Error returns are:

{ "Job Completed with No Error", "Not Supported", "Unknown", "Timeout", "Failed", "Invalid Parameter", "In Use", "DMTF Reserved", "Method Parameters Checked - Job Started", "Method Reserved", "Vendor Specific" }}

25.6.3 Deallocate

Starts a job to remove remote resources (from the AllocatedResources collection) and return them to the RemoteResources collection.

Deallocate (

[IN, Description (Enumeration indicating the type of element being deallocated. This type value shall match the type of the instances.),

ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8" },

Values { "Unknown", "Reserved", "Any Type", "StorageVolume", "StorageExtent", "StoragePool", "ComputerSystem", "LogicalDisk", "FileShare" }}

Only "3" (StorageVolume) is supported in this version of SMI-S.

uint16 **ElementType**;

[IN (false), OUT, Description (Reference to the job (may be null if job completed).)]

CIM_ConcreteJob REF **Job**,

[IN, Description (The reference to the AllocatedResource collection from which Elements are being removed.)]

SNIA_AllocatedResources REF **Collection**,

[IN, Description (Array of strings containing representations of references to CIM_ManagedElement instances, that are being deallocated from the AllocatedResources Collection.)]

string **InElements[]**;

Error returns are:

{ "Job Completed with No Error", "Not Supported", "Unknown", "Timeout", "Failed", "Invalid Parameter", "In Use", "DMTF Reserved", "Method Parameters Checked - Job Started", "Method Reserved", "Vendor Specific" }

25.7 Client Considerations and Recipes

25.7.1 Recipe MPCP01: Determining Resources used by cascading Profiles

This recipe is not defined in this International Standard. It will be included in a future revision, based on implementation experience.

25.7.2 Recipe MPCP02: Monitoring the existence of Cascading Profiles

This recipe is not defined in this International Standard. It will be included in a future revision, based on implementation experience.

25.7.3 OPTIONAL: Recipe MPCP03: Allocation of Leaf Resources

This recipe is not defined in this International Standard. It will be included in a future revision, based on implementation experience.

25.7.4 OPTIONAL: Recipe MPCP04: Deallocation of Leaf Resources

This recipe is not defined in this International Standard. It will be included in a future revision, based on implementation experience.

25.7.5 Recipe MPCP05: Monitoring the existence of "Stitching" between Profiles

This recipe is not defined in this International Standard. It will be included in a future revision, based on implementation experience.

25.7.6 Supported SNIA_CascadingCapabilities Patterns

The SNIA_CascadingCapabilities patterns in Table 250 are formally recognized and supported by this version of SMI-S.

Table 250 - Cascading Capabilities Patterns

FeaturesSupported	SupportedElementTypes	SynchronousMethods Supported	AsynchronousMethods Supported
none	StorageVolume	none	none
Ownership, Leaf Credentials, OM Dependencies, Allocation Service	StorageVolume	Allocation Deallocation	Allocation Deallocation
Allocation Service	StorageVolume	Allocation Deallocation	none
Allocation Service	StorageVolume	none	Allocation Deallocation
Ownership, Leaf Credentials, OM Dependencies	StorageVolume	none	none

25.8 Registered Name and Version

Cascading version 1.3.0 (Component Profile).

25.9 CIM Elements

25.9.1 General

Table 251 describes the CIM elements for Cascading.

Table 251 - CIM Elements for Cascading

Element Name	Requirement	Description
25.9.2 CIM_ComputerSystem (Leaf System)	Mandatory	'Top level' system that represents the leaf system.
25.9.3 CIM_Dependency (Object Managers)	Conditional	Conditional requirement: This is required if Leaf ObjectManagers are modeled. This associates the Object Manager of the Leaf System to the Object Manager of the Cascading System.
25.9.4 CIM_Dependency (Profile to Object Manager)	Conditional	Conditional requirement: This is required if RegisteredProfiles of Leaf systems are modeled. This associates the RegisteredProfile of a leaf system to the Object Manager of the leaf system.
25.9.5 CIM_Dependency (Systems)	Mandatory	This associates the Leaf System to the Cascading System.
25.9.6 CIM_ElementCapabilities	Mandatory	This associates the CascadingCapabilities to the cascading system (e.g., ComputerSystem).
25.9.7 CIM_ElementConformsToProfile (Leaf)	Conditional	Conditional requirement: This is required if RegisteredProfiles of Leaf systems are modeled. This associates the RegisteredProfile of the Leaf Profile to the Leaf system (e.g., ComputerSystem).
25.9.8 CIM_HostedCollection (Allocated Resources)	Mandatory	This would associate the AllocatedResources collection to the top level system for the Cascading Profile (e.g., Storage Virtualizer).
25.9.9 CIM_HostedCollection (Remote Resources)	Conditional	Conditional requirement: This is required if SNIA_RemoteResources is modeled. This would associate the RemoteResources collection to the top level system for the Cascading Profile (e.g., Storage Virtualizer).
25.9.10 CIM_HostedService (Allocation Service)	Conditional	Conditional requirement: This is required if SNIA_AllocationService is modeled. This associates the AllocationService to the system in the cascading profile that hosts the service.
25.9.11 CIM_HostedService (Object Manager)	Conditional	Conditional requirement: This is required if Leaf ObjectManagers are modeled. This associates the ObjectManager to the system in the cascading profile that hosts the service.
25.9.12 CIM_LogicalDisk	Conditional	Conditional requirement: This is required if SNIA_CascadingCapabilities.SupportedElementTypes = '7' ('LogicalDisk'). A remote LogicalDisk that is imported to the referencing profile.
25.9.13 CIM_LogicalIdentity (General)	Mandatory	Associates local resource (e.g., StorageExtent) to a remote (imported) resource (e.g., StorageVolume or LogicalDisk).

Table 251 - CIM Elements for Cascading

Element Name	Requirement	Description
25.9.14 CIM_LogicalIdentity (LogicalDisk)	Conditional	Conditional requirement: This is required if SNIA_CascadingCapabilities.SupportedElementTypes = '7' ('LogicalDisk'). Associates local StorageExtent to a remote (imported) LogicalDisk.
25.9.15 CIM_LogicalIdentity (StorageVolume)	Conditional	Conditional requirement: This is required if SNIA_CascadingCapabilities.SupportedElementTypes = '3' ('StorageVolume'). Associates local StorageExtent to a remote (imported) StorageVolume.
25.9.16 CIM_MemberOfCollection (Allocated Resources)	Mandatory	This supports collecting leaf resources. This is required to support the AllocatedResources collection.
25.9.17 CIM_MemberOfCollection (Remote Resources)	Conditional	Conditional requirement: This is required if SNIA_RemoteResources is modeled. This supports collecting leaf resources. This is optional when used to support the RemoteResources collection (the RemoteResources collection is optional).
25.9.18 CIM_Namespace (Leaf)	Conditional	Conditional requirement: This is required if Leaf ObjectManagers are modeled. There would be one for every namespace supported.
25.9.19 CIM_NamespaceInManager (Leaf)	Conditional	Conditional requirement: This is required if Leaf ObjectManagers are modeled. This associates the namespace to the ObjectManager.
25.9.20 CIM_ObjectManager (Leaf)	Optional	This is the Object Manager service of the CIM Server.
25.9.21 CIM_RegisteredProfile (Leaf)	Optional	A registered profile that is supported by the CIM Server.
25.9.22 CIM_RemoteServiceAccessPoint (Leaf)	Optional	CIM_RemoteServiceAccessPoint represents the management interface to a leaf system.
25.9.23 CIM_SAPAvailableForElement	Conditional	Conditional requirement: This is required if CIM_RemoteServiceAccessPoint is modeled. Represents the association between a RemoteServiceAccessPoint and the leaf System to which it provides access.
25.9.24 CIM_StorageVolume	Conditional	Conditional requirement: This is required if SNIA_CascadingCapabilities.SupportedElementTypes = '3' ('StorageVolume'). A remote StorageVolume that is imported to the referencing profile.
25.9.25 CIM_SystemDevice (Leaf Devices)	Conditional	Conditional requirement: This is required if SNIA_CascadingCapabilities.SupportedElementTypes = '3' '4' '7' (StorageVolume StorageExtent LogicalDisk). This association links LogicalDevice remote resources to the scoping system. This is used to associate the remote resources with the System that manages them.
25.9.26 SNIA_AllocatedResources	Mandatory	This is a SystemSpecificCollection for collecting leaf resources that have been deployed for use in the Cascading profile (e.g., StorageVolumes assigned to a virtualizer's StoragePool).

Table 251 - CIM Elements for Cascading

Element Name	Requirement	Description
25.9.27 SNIA_AllocationService	Optional	This service provides methods for allocating and deallocating leaf resources.
25.9.28 SNIA_CascadingCapabilities	Mandatory	This defines the cascading capabilities supported by the implementation of the profile.
25.9.29 SNIA_RemoteResources	Optional	This is a SystemSpecificCollection for collecting leaf resources that may be allocated by the system of the Cascading profile (e.g., StorageVolumes assigned to a virtualizer's StoragePool).

25.9.2 CIM_ComputerSystem (Leaf System)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 252 describes class CIM_ComputerSystem (Leaf System).

Table 252 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Leaf System)

Properties	Flags	Requirement	Description and Notes
CreationClassName		Mandatory	
Name		Mandatory	Unique identifier for the leaf system. E.g., IP address.
ElementName		Mandatory	User friendly name.
OtherIdentifyingInfo	C	Mandatory	
IdentifyingDescriptions	C	Mandatory	
OperationalStatus		Mandatory	Overall status of the Leaf system.
NameFormat		Mandatory	Format for Name property.
Dedicated		Mandatory	Indicates that this computer system is dedicated to operation as a leaf system.
PrimaryOwnerContact	M	Optional	Contact a details for owner.
PrimaryOwnerName	M	Optional	Owner of the Leaf system.

25.9.3 CIM_Dependency (Object Managers)

CIM_Dependency is an association between an Object Manager of a Leaf System and the Object Manager of the Cascading System (ComputerSystem). If the Leaf System and the Cascading System are supported by the same Object Manager, then no Dependency would exist.

CIM_Dependency is not subclassed from anything.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if Leaf ObjectManagers are modeled.

Table 253 describes class CIM_Dependency (Object Managers).

Table 253 - SMI Referenced Properties/Methods for CIM_Dependency (Object Managers)

Properties	Flags	Requirement	Description and Notes
Antecedent		Mandatory	The Object Manager of the Cascading System.
Dependent		Mandatory	The Object Manager of the Leaf System.

25.9.4 CIM_Dependency (Profile to Object Manager)

CIM_Dependency is an association between RegisteredProfile and the Object Manager that provides the management interface.

CIM_Dependency is not subclassed from anything.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if RegisteredProfiles of Leaf systems are modeled.

Table 254 describes class CIM_Dependency (Profile to Object Manager).

Table 254 - SMI Referenced Properties/Methods for CIM_Dependency (Profile to Object Manager)

Properties	Flags	Requirement	Description and Notes
Antecedent		Mandatory	The Leaf Object Manager.
Dependent		Mandatory	The RegisteredProfile for the Leaf System.

25.9.5 CIM_Dependency (Systems)

CIM_Dependency is an association between a Leaf System and the Cascading System (ComputerSystem). The specific nature of the dependency is determined by associations between resources of the cascading system and resources of the leaf system.

CIM_Dependency is not subclassed from anything.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 255 describes class CIM_Dependency (Systems).

Table 255 - SMI Referenced Properties/Methods for CIM_Dependency (Systems)

Properties	Flags	Requirement	Description and Notes
Antecedent		Mandatory	The Cascading System.
Dependent		Mandatory	The Leaf System.

25.9.6 CIM_ElementCapabilities

CIM_ElementCapabilities represents the association between ManagedElements (i.e., ComputerSystem) and their capabilities (e.g., SNIA_CascadingCapabilities).

CIM_ElementCapabilities is not subclassed from anything.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 256 describes class CIM_ElementCapabilities.

Table 256 - SMI Referenced Properties/Methods for CIM_ElementCapabilities

Properties	Flags	Requirement	Description and Notes
ManagedElement		Mandatory	
Capabilities		Mandatory	

25.9.7 CIM_ElementConformsToProfile (Leaf)

CIM_ElementConformsToProfile is the association between the RegisteredProfile of the leaf profile and the system of the leaf (i.e., leaf ComputerSystem).

CIM_ElementConformsToProfile is not subclassed from anything.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if RegisteredProfiles of Leaf systems are modeled.

Table 257 describes class CIM_ElementConformsToProfile (Leaf).

Table 257 - SMI Referenced Properties/Methods for CIM_ElementConformsToProfile (Leaf)

Properties	Flags	Requirement	Description and Notes
ConformantStandard		Mandatory	The RegisteredProfile of the leaf system.
ManagedElement		Mandatory	Reference to the top-level system of the leaf profile.

25.9.8 CIM_HostedCollection (Allocated Resources)

CIM_HostedCollection defines a SystemSpecificCollection in the context of a scoping System. It represents a Collection that only has meaning in the context of a System, and/or whose elements are restricted by the definition of the System. In the Cascading Subprofile, it is used to associate the Allocated Resources to the top level Computer System of the Cascading Profile.

CIM_HostedCollection is subclassed from CIM_HostedDependency.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 258 describes class CIM_HostedCollection (Allocated Resources).

Table 258 - SMI Referenced Properties/Methods for CIM_HostedCollection (Allocated Resources)

Properties	Flags	Requirement	Description and Notes
Antecedent		Mandatory	
Dependent		Mandatory	

25.9.9 CIM_HostedCollection (Remote Resources)

CIM_HostedCollection defines a SystemSpecificCollection in the context of a scoping System. It represents a Collection that only has meaning in the context of a System, and/or whose elements are restricted by the definition of the System. In the Cascading Subprofile, it is used to associate the Remote Resources to the top level Computer System of the Cascading Profile.

CIM_HostedCollection is subclassed from CIM_HostedDependency.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if SNIA_RemoteResources is modeled.

Table 259 describes class CIM_HostedCollection (Remote Resources).

Table 259 - SMI Referenced Properties/Methods for CIM_HostedCollection (Remote Resources)

Properties	Flags	Requirement	Description and Notes
Antecedent		Mandatory	
Dependent		Mandatory	

25.9.10 CIM_HostedService (Allocation Service)

CIM_HostedService is an association between a Service (SNIA_AllocationService) and the System (ComputerSystem) on which the functionality resides.

CIM_HostedService is subclassed from CIM_HostedDependency.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if SNIA_AllocationService is modeled.

Table 260 describes class CIM_HostedService (Allocation Service).

Table 260 - SMI Referenced Properties/Methods for CIM_HostedService (Allocation Service)

Properties	Flags	Requirement	Description and Notes
Dependent		Mandatory	
Antecedent		Mandatory	>The AllocationService hosted on the System.

25.9.11 CIM_HostedService (Object Manager)

CIM_HostedService is an association between a Service (SNIA_AllocationService) and the System (ComputerSystem) on which the functionality resides.

CIM_HostedService is subclassed from CIM_HostedDependency.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if Leaf ObjectManagers are modeled.

Table 261 describes class CIM_HostedService (Object Manager).

Table 261 - SMI Referenced Properties/Methods for CIM_HostedService (Object Manager)

Properties	Flags	Requirement	Description and Notes
Dependent		Mandatory	
Antecedent		Mandatory	

25.9.12 CIM_LogicalDisk

A remote LogicalDisk that is imported to the referencing profile. If the referencing profile has access to the leaf profile, the data in this class should reflect what the referencing profile obtains from that profile. If the referencing profile does not have access to the leaf profile, then this should be filled out as best can be done.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if SNIA_CascadingCapabilities.SupportedElementTypes = '7' ('LogicalDisk').

Table 262 describes class CIM_LogicalDisk.

Table 262 - SMI Referenced Properties/Methods for CIM_LogicalDisk

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
DeviceID		Mandatory	Opaque identifier.
ElementName		Optional	User-friendly name.
Name		Mandatory	OS Device Name.
NameFormat		Mandatory	Format for name.
ExtentStatus		Mandatory	
OperationalStatus		Mandatory	Value shall be 2 3 6 8 15 (OK or Degraded or Error or Starting or Dormant).
BlockSize		Mandatory	
NumberOfBlocks		Mandatory	The number of blocks of capacity consumed from the parent StoragePool.

Table 262 - SMI Referenced Properties/Methods for CIM_LogicalDisk

Properties	Flags	Requirement	Description and Notes
ConsumableBlocks		Mandatory	The number of blocks usable by consumers.
IsBasedOnUnderlyingRedundancy		Mandatory	
NoSinglePointOfFailure		Mandatory	
DataRedundancy		Mandatory	
PackageRedundancy		Mandatory	
DeltaReservation		Mandatory	
Usage		Optional	The specialized usage intended for this element.
OtherUsageDescription		Optional	Set when Usage value is "Other".
ClientSettableUsage		Optional	Lists Usage values that can be set by a client for this element.
Caption	N	Optional	Not Specified in this version of the Profile.
Description	N	Optional	Not Specified in this version of the Profile.
InstallDate	N	Optional	Not Specified in this version of the Profile.
StatusDescriptions	N	Optional	Not Specified in this version of the Profile.
HealthState	N	Optional	Not Specified in this version of the Profile.
EnabledState	N	Optional	Not Specified in this version of the Profile.
OtherEnabledState	N	Optional	Not Specified in this version of the Profile.
RequestedState	N	Optional	Not Specified in this version of the Profile.
EnabledDefault	N	Optional	Not Specified in this version of the Profile.
TimeOfLastStateChange	N	Optional	Not Specified in this version of the Profile.

25.9.13 CIM_LogicalIdentity (General)

Associates local resource (e.g., StorageExtent) to a remote (imported) resource (e.g., StorageVolume or LogicalDisk).

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 263 describes class CIM_LogicalIdentity (General).

Table 263 - SMI Referenced Properties/Methods for CIM_LogicalIdentity (General)

Properties	Flags	Requirement	Description and Notes
SystemElement		Mandatory	This is a reference to the remote (imported) resource.
SameElement		Mandatory	This is a reference to the local resource that maps to the remote (imported) resource.

25.9.14 CIM_LogicalIdentity (LogicalDisk)

Associates local StorageExtent to a remote (imported) LogicalDisk.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if SNIA_CascadingCapabilities.SupportedElementTypes = '7' ('LogicalDisk').

Table 264 describes class CIM_LogicalIdentity (LogicalDisk).

Table 264 - SMI Referenced Properties/Methods for CIM_LogicalIdentity (LogicalDisk)

Properties	Flags	Requirement	Description and Notes
SystemElement		Mandatory	This is a reference to the remote (imported) LogicalDisk.
SameElement		Mandatory	This is a reference to the local StorageExtent that maps to the remote (imported) LogicalDisk.

25.9.15 CIM_LogicalIdentity (StorageVolume)

Associates local StorageExtent to a remote (imported) StorageVolume.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if SNIA_CascadingCapabilities.SupportedElementTypes = '3' ('StorageVolume').

Table 265 describes class CIM_LogicalIdentity (StorageVolume).

Table 265 - SMI Referenced Properties/Methods for CIM_LogicalIdentity (StorageVolume)

Properties	Flags	Requirement	Description and Notes
SystemElement		Mandatory	This is a reference to the remote (imported) StorageVolume.
SameElement		Mandatory	This is a reference to the local StorageExtent that maps to the remote (imported) StorageVolume.

25.9.16 CIM_MemberOfCollection (Allocated Resources)

This use of MemberOfCollection is to collect all resource instances (in the AllocatedResources collection). Each association is created as a result of the Allocate method or as a side effect of a cascading profile specific operation.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 266 describes class CIM_MemberOfCollection (Allocated Resources).

Table 266 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Allocated Resources)

Properties	Flags	Requirement	Description and Notes
Member		Mandatory	
Collection		Mandatory	

25.9.17 CIM_MemberOfCollection (Remote Resources)

This use of MemberOfCollection is to collect all resource instances (in the RemoteResources collection). Each association (and the RemoteResources collection, itself) is created through external means.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if SNIA_RemoteResources is modeled.

Table 267 describes class CIM_MemberOfCollection (Remote Resources).

Table 267 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (Remote Resources)

Properties	Flags	Requirement	Description and Notes
Member		Mandatory	
Collection		Mandatory	

25.9.18 CIM_Namespace (Leaf)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if Leaf ObjectManagers are modeled.

Table 268 describes class CIM_Namespace (Leaf).

Table 268 - SMI Referenced Properties/Methods for CIM_Namespace (Leaf)

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
ObjectManagerCreationClassName		Mandatory	
ObjectManagerName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	
ClassType		Mandatory	

Table 268 - SMI Referenced Properties/Methods for CIM_Namespace (Leaf)

Properties	Flags	Requirement	Description and Notes
DescriptionOfClassType		Mandatory	Mandatory if ClassType is set to 'Other'.
ClassInfo		Optional	Deprecated in the MOF, but required for 1.0 compatibility.
DescriptionOfClassInfo		Optional	Deprecated in the MOF, but mandatory for 1.0 compatibility. Mandatory if ClassInfo is set to 'Other'.

25.9.19 CIM_NamespaceInManager (Leaf)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if Leaf ObjectManagers are modeled.

Table 269 describes class CIM_NamespaceInManager (Leaf).

Table 269 - SMI Referenced Properties/Methods for CIM_NamespaceInManager (Leaf)

Properties	Flags	Requirement	Description and Notes
Antecedent		Mandatory	
Dependent		Mandatory	

25.9.20 CIM_ObjectManager (Leaf)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 270 describes class CIM_ObjectManager (Leaf).

Table 270 - SMI Referenced Properties/Methods for CIM_ObjectManager (Leaf)

Properties	Flags	Requirement	Description and Notes
Name		Mandatory	
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
ElementName		Mandatory	
Description		Mandatory	
OperationalStatus		Mandatory	
Started		Mandatory	
StopService()		Optional	

25.9.21 CIM_RegisteredProfile (Leaf)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 271 describes class CIM_RegisteredProfile (Leaf).

Table 271 - SMI Referenced Properties/Methods for CIM_RegisteredProfile (Leaf)

Properties	Flags	Requirement	Description and Notes
InstanceID		Mandatory	This is a unique value for the profile instance.
RegisteredOrganization		Mandatory	This is the official name of the organization that created the Profile. For SMI-S profiles, this would be SNIA.
OtherRegisteredOrganization		Optional	
RegisteredName		Mandatory	This is the name assigned by the organization that created the profile.
RegisteredVersion		Mandatory	This is the version number of the organization that defined the profile.
AdvertiseTypes		Mandatory	Defines the advertisement of this profile. If the property is null then no advertisement is defined. A value of 1 is used to indicate 'other' and a 3 is used to indicate 'SLP'.
AdvertiseTypeDescriptions		Optional	This shall not be NULL if 'Other' is identified in AdvertiseType.

25.9.22 CIM_RemoteServiceAccessPoint (Leaf)

CIM_RemoteServiceAccessPoint is an instance that provides access information for accessing the actual leaf profile via a management interface.

CIM_RemoteServiceAccessPoint is not subclassed from CIM_ServiceAccessPoint.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 272 describes class CIM_RemoteServiceAccessPoint (Leaf).

Table 272 - SMI Referenced Properties/Methods for CIM_RemoteServiceAccessPoint (Leaf)

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	The CIM Class name of the Computer System hosting the management interface.
SystemName		Mandatory	The name of the Computer System hosting the management interface.
CreationClassName		Mandatory	The CIM Class name of the management interface.
Name		Mandatory	The unique name of the management interface.

25.9.23 CIM_SAPAvailableForElement

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if CIM_RemoteServiceAccessPoint is modeled.

Table 273 describes class CIM_SAPAvailableForElement.

Table 273 - SMI Referenced Properties/Methods for CIM_SAPAvailableForElement

Properties	Flags	Requirement	Description and Notes
ManagedElement		Mandatory	Leaf System.
AvailableSAP		Mandatory	

25.9.24 CIM_StorageVolume

A remote StorageVolume that is imported to the referencing profile. If the referencing profile has access to the leaf profile, the data in this class should reflect what the referencing profile obtains from that profile. If the referencing profile does not have access to the leaf profile, then this should be filled out as best can be done.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if SNIA_CascadingCapabilities.SupportedElementTypes = '3' ('StorageVolume').

Table 274 describes class CIM_StorageVolume.

Table 274 - SMI Referenced Properties/Methods for CIM_StorageVolume

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
DeviceID		Mandatory	Opaque identifier.
ElementName		Optional	User-friendly name.
Name	CD	Mandatory	The identifier for this volume.
OtherIdentifyingInfo	CD	Optional	Additional correlatable names.
IdentifyingDescriptions		Optional	
NameFormat		Mandatory	The type of identifier in the Name property.
NameNamespace		Mandatory	The namespace that defines uniqueness for the NameFormat.
ExtentStatus		Mandatory	
OperationalStatus		Mandatory	Value shall be 2 3 6 8 15 (OK or Degraded or Error or Starting or Dormant).
BlockSize		Mandatory	
NumberOfBlocks		Mandatory	The number of blocks of capacity consumed from the parent StoragePool.
ConsumableBlocks		Mandatory	The number of blocks usable by consumers.
IsBasedOnUnderlyingRedundancy		Mandatory	

Table 274 - SMI Referenced Properties/Methods for CIM_StorageVolume

Properties	Flags	Requirement	Description and Notes
NoSinglePointOfFailure		Mandatory	
DataRedundancy		Mandatory	
PackageRedundancy		Mandatory	
DeltaReservation		Mandatory	
Usage		Optional	The specialized usage intended for this element.
OtherUsageDescription		Optional	Set when Usage value is "Other".
ClientSettableUsage		Optional	Lists Usage values that can be set by a client for this element.
Caption	N	Optional	Not Specified in this version of the Profile.
Description	N	Optional	Not Specified in this version of the Profile.
InstallDate	N	Optional	Not Specified in this version of the Profile.
StatusDescriptions	N	Optional	Not Specified in this version of the Profile.
HealthState	N	Optional	Not Specified in this version of the Profile.
EnabledState	N	Optional	Not Specified in this version of the Profile.
OtherEnabledState	N	Optional	Not Specified in this version of the Profile.
RequestedState	N	Optional	Not Specified in this version of the Profile.
EnabledDefault	N	Optional	Not Specified in this version of the Profile.
TimeOfLastStateChange	N	Optional	Not Specified in this version of the Profile.

25.9.25 CIM_SystemDevice (Leaf Devices)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: This is required if SNIA_CascadingCapabilities.SupportedElementTypes = '3'|'4'|'7' (StorageVolume | StorageExtent | LogicalDisk).

Table 275 describes class CIM_SystemDevice (Leaf Devices).

Table 275 - SMI Referenced Properties/Methods for CIM_SystemDevice (Leaf Devices)

Properties	Flags	Requirement	Description and Notes
GroupComponent		Mandatory	The Leaf Computer System that contains this device.
PartComponent		Mandatory	The logical device that is managed by a computer system.

25.9.26 SNIA_AllocatedResources

An instance of a default SNIA_AllocatedResources defines the set of remote (leaf) resources that are allocated and in use by the Cascading Profile.

SNIA_AllocatedResources is subclassed from CIM_SystemSpecificCollection.

At least one instance of the SNIA_AllocatedResources shall exist for a Profile and shall be hosted by one of the ComputerSystems of that Profile.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 276 describes class SNIA_AllocatedResources.

Table 276 - SMI Referenced Properties/Methods for SNIA_AllocatedResources

Properties	Flags	Requirement	Description and Notes
InstanceID		Mandatory	
ElementName		Mandatory	A user-friendly name for the AllocatedResources collection (e.g., AllocatedVolumes).
ElementType		Mandatory	The type of remote resources collected by the AllocatedResources collection. For this version of SMI-S, the only value supported is '3' (StorageVolume).

25.9.27 SNIA_AllocationService

The SNIA_AllocationService class provides methods for allocating and deallocating remote resources for use in the Cascading Profile.

The SNIA_AllocationService class is subclassed from CIM_Service.

There may be an instance of the SNIA_AllocationService if Allocation or Deallocation are supported.

The methods that are supported can be determined from the SynchronousMethodsSupported and AsynchronousMethodsSupported properties of the SNIA_CascadingCapabilities.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 277 describes class SNIA_AllocationService.

Table 277 - SMI Referenced Properties/Methods for SNIA_AllocationService

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	
Allocate()		Optional	Support for this method is optional. This method allocates remote (leaf) resources to the AllocatedResources collection.
Deallocate()		Optional	Support for this method is optional. This method is used to remove remote (leaf) resources from the AllocatedResources collection.

25.9.28 SNIA_CascadingCapabilities

An instance of the SNIA_CascadingCapabilities class defines the specific support provided with the implementation of the Cascading Profile.

There would be zero or one instance of this class in a profile. There would be none if the profile did not support the Cascading Subprofile. There would be exactly one instance if the profile did support the Cascading Subprofile.

SNIA_CascadingCapabilities class is subclassed from CIM_Capabilities.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 278 describes class SNIA_CascadingCapabilities.

Table 278 - SMI Referenced Properties/Methods for SNIA_CascadingCapabilities

Properties	Flags	Requirement	Description and Notes
InstanceID		Mandatory	
ElementName		Mandatory	
FeaturesSupported		Mandatory	ValueMap { '2', '3', '4', '5' }, Values {'Ownership', 'Leaf Credentials', 'OM Dependencies', 'Allocation Service'}.
SupportedElementTypes		Mandatory	For this version of SMI-S, only the value '3' (StorageVolume) is supported. ValueMap { '2', '3', '4', '5', '6', '7', '8' }, Values {'Any Type', 'StorageVolume', 'StorageExtent', 'StoragePool', 'ComputerSystem', 'LogicalDisk', 'FileShare'}.
SupportedSynchronousActions		Mandatory	ValueMap { '2', '3' }, Values {'Allocation', 'Deallocation'}.
SupportedAsynchronousActions		Mandatory	ValueMap { '2', '3' }, Values {'Allocation', 'Deallocation'}.

25.9.29 SNIA_RemoteResources

An instance of a default SNIA_RemoteResources defines the set of remote (leaf) resources that are available to be used by the Cascading Profile.

SNIA_RemoteResources is subclassed from CIM_SystemSpecificCollection.

One instance of the SNIA_RemoteResources would exist for each Element type for a Profile and shall be hosted by one of the ComputerSystems of that Profile.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 279 describes class SNIA_RemoteResources.

Table 279 - SMI Referenced Properties/Methods for SNIA_RemoteResources

Properties	Flags	Requirement	Description and Notes
InstanceID		Mandatory	
ElementName		Mandatory	A user-friendly name for the RemoteResources collection (e.g., RemoteVolumes).
ElementType		Mandatory	The type of remote resources collected by the RemoteResources collection. For this version of SMI-S, the only value supported is '3' (StorageVolume).

DEPRECATED

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

STABLE

26 Health Package

26.1 Description

26.1.1 General

Failures and abnormal occurrences are a common and expected part of monitoring, controlling, and configuring devices and applications. A SMI-S client needs to be prepared at all times to trap unexpected situations and take appropriate action. This package defines the general mechanisms used in the expression of health in SMI-S. This package does not define the particular way a particular profile, subprofile, or package reports health.

This package builds on *Information Technology – Storage Management, Part 2 Common Architecture* 6 Health and Fault Management. In particular, this package defines the basis of all the clauses that currently and will exist in this International Standard or future versions of same.

26.1.2 Error Reporting Mechanism

Error are reports for many reasons. Not all the reasons are directly related to the operation being imposed on the implementation by the client. It is therefore necessary for the client to be able to distinguish between errors that are associated to problems in the formation and invocation of a method, extrinsic or intrinsic, or are related to other conditions.

The client application may need to reform the method call itself, by fixing parameters for example, or the client may need to stop what it is attempting. At a basic level, the client needs to know that this operation will succeed at all, given the prevailing conditions on the managed element. A client may also need to notify the end-user of the situation that is preventing the client from fulfilling its function. A HFM application may need to investigate the failure and develop a prognosis.

The types of errors are categorized in the three following types.

- a) Errors associated to the method call.
- b) Errors caused by adverse prevailing conditions in the managed element.
- c) Errors caused by adverse prevailing conditions in the WBEM Server or related, infrastructural components.

Obviously, the method called may not exist. There may be a spelling mistake for the method name. One or more of the parameters may be incorrectly formed, expressed, or otherwise invalid. The first type of error, type a, is designed to inform the client that the operation attempted is still valid, but that the request was faulty. The intent of such an error is to tell the client what is wrong with the method call and allow the method to be invoked again.

On the other hand, the device or application may be in some failure condition which prevents it from honoring this particular or several method calls. This type of error, type b, tells the client that it is unlikely that the method being attempted will be honored. Specifically, the method execution is blocked by the prevailing condition being described in the error itself. Given the presence of both type a and type b error situations, the implementation should report the type b error. In this case, it does not matter how many fixes are made to the method call, the method call will fail anyway.

The WBEM Service is a separate architectural element from the managed element itself. It can fail, even though the methods and the managed element itself are without error. For example, the WBEM Server may allow only a limited number of concurrent connection or request and reject all others. The server may be shutting down or starting up and thus be unable to process any requests at the time. Unlike type b errors, type c errors are usually transient in nature. Since a failure in the WBEM Server or its components constitutes a communications failure, the reporting of type c errors shall take precedence over all other existing error type conditions.

The WBEM Server returns a error response or a results response to the request, which contains the operation previously mentioned. Errors in WBEM may be reported through two ways. The status code itself provides basic failure information. The number of status codes is very limited. Also on conveyed on the error response, is a Error instance. The Error provides vastly most information than the status code and, as such, is a superior mechanism for reporting errors.

The CIM Error provides attributes to express the categorization and severity of the error. More importantly, the CIM Error and AlertIndication, to be discussed later, contain the exact expression of the nature of the error and additional parameters to that error.

26.1.3 Event Reporting Mechanism

It is not sufficient to simply report the adverse conditions of the device or application through the error reporting mechanism. Many of the adverse conditions that would be reported to a client application attempting control or configuration operations are also of interest to client applications monitoring the very same device or application.

The CIM Event model provides a special class for reporting event conditions, AlertIndication. The AlertIndication is used to report a device or application conditions that may also be represented in one or more other instances. When the implementation detects the presence of a supported condition, it generates an AlertIndication to those listening clients.

It is recommended that the type b and type c errors reported in also be reported through AlertIndications.

26.1.4 Standard Events

The expression of Error or an Alertindication is not entirely meaningful to the SMI-S client without the standardization. A client can use these classes to determine the category, severity, and some other characteristics of the event, but the client can not determine the exact nature of the event without this standardization.

Standard events are registered and this registry is maintained by some organization or company, like SNIA.

Primary event identification and characterization properties:

- **OwningEntity**
This property defines the registration entity for the event. The entities that are in scope for SMI-S are "DMTF" and "SNIA". If the OwningEntity is neither of these, then this International Standard provides no meaning for this event.
- **MessageID**
This property defines an event identifier that is unique for the OwningEntity. The combination of the OwningEntity and MessageID defines the entry in the registry.
- **Message**
This property contains the message that can be forwarded to the end-user. The message is built from using the static, MessageFormatString, and dynamic, MessageArguments, components. This text may be localized. This text is not intended for programmatic processing
- **MessageArguments**
This property defines the variable content for the message. The client would programmatically process the arguments to get further details on the nature of the event. For example, the message argument can tell the client which method parameter has a problem and what the problem is.
- **MessageFormatString**
This property defines the static component of the message. This property is not included in the event instance itself and is only present in the event registry.

26.1.5 Reporting Health

Many devices or applications can attempt to fix themselves upon encountering some adverse condition. The set of components which the device or application can attempt to fix is called the Fault Region. The set may include part or all of other devices or applications. Having the Fault Regions declared helps a HFM application, acting as a doctor, to do no harm by attempting to interfere and thereby adversely effect the corrective action being attempted.

When components fail or become degraded, they can cause other components to fail or become degraded. For an HFM application to report or attempt to diagnose the problem, the device or application should express what the cause and effect relationships are that define the extent of the components affected by the failure or degradation. The RelatedElementCausingError class provides just such a mechanism.

The cause and effect relationships identified by the RelatedElementCausingError association may be a chain of cause and effect relationships with many levels. Given that devices or applications are sometimes subject to several levels of decomposition, each level of may have its own set of these associations that represent the ranking of cause and effect relationships and their effect on the parent component on the given level.

26.1.6 Computer System Operational Status

For most profiles, the ComputerSystem class is used to define the top or head of the object hierarchy. A profile may allow for partitioning or clustering by having more than one ComputerSystem, but one ComputerSystem often represents the device or application representation. In this role, it is important that the summary of the health of the device or application be declared in the ComputerSystem instance.

Table 280 - OperationalStatus Details

Primary Operational Status	Subsidiary Operational Status	Description
2 "OK"		The system has a good status.
2 "OK"	4 "Stressed"	The system is stressed, for example the temperature is over limit or there is too much IO in progress.
2 "OK"	5 "Predictive Failure"	The system will probably fail sometime soon.
3 "Degraded"		The system is operational but not at 100% redundancy. A component has suffered a failure or something is running slow.
6 "Error"		An error has occurred causing the system to stop. This error may be recoverable with operator intervention.
6 "Error"	7 "Non-recoverable error"	A severe error has occurred. Operator intervention is unlikely to fix it.
6 "Error"	16 "Supporting entity in error"	A modeled element has failed.
12 "No contact"		The provider knows about the array but has not talked to it since last reboot.
13 "Lost communication"		The provider used to be able to communicate with the array, but has now lost contact.
8 "Starting"		The system is starting up.
9 "Stopping"		The system is shutting down.
10 "Stopped"		The data path is OK but shut down, the management channel is still working.

OperationalStatus is an array. The primary and subsidiary statuses are both OperationalStatus property, and are summarized in Table 280. If the subsidiary operational status is present in the array, it is intended to provide additional clarification to the primary operational status. The implementation shall report one of the above combinations of statuses. It may also report additional statuses beyond the ones defined in Table 280.

The operational status combinations listed in Table 280 that include descriptions about “provider” (i.e., the CIM Provider), are only valid in those cases where the implementation of SMI-S employs a proxy provider.

The operational statuses listed in Table 280 shall not be used to report the status of the WBEM Server itself.

EXPERIMENTAL

26.1.7 Event Reporting

The implementation may report Event or AlertIndication instances. The profile, subprofile, or package that includes this package defines whether or not these events are supported and when the events are produced.

If the support Event or AlertIndication is implemented, then the implementation shall also support the common messages through both Errors and AlertIndications. This means that the implementation produce the common event listed in the registry when the condition, also described in the registry, is present.

It is mandatory to report error conditions through both AlertIndication or Lifecycle indication and Error in those cases where Error is returned when the method call failed for reasons other than the method call itself. For example, if the device is over heated, then a method call can fail because of this condition. It is expected that the device will report an over heat AlertIndication to listening clients as well.

26.1.8 Fault Region

If the device or application is itself attempting to rectify an adverse condition reported through a standard error, then the implementation shall report what corrective action, if any, it is taking. This is necessary to prevent a HFM application from also trying to rectify the very same condition. An HFM application should avoid a interfering with ongoing corrective action taken by the device or application itself.

The corrective action may be a process, like hardware diagnostics or volume rebuild. In which case, the above requirement is fulfilled by expressing the instances representing the process.

The corrective action may be a state change, like reboot. In which case, the above requirement is fulfilled by expressing the state change in some CIM Instances.

In all cases, the profile, subprofile, or package that includes this package defines the standard events included and the associated, possible corrective actions taken in response to these events.

26.1.9 RelatedElementCausingError

This package provides a mechanism in which the effect of a component failure on other components can be reported. the RelatedElementCausingError association defines what components are causing a particular component to failure or become degraded.

Some effects are more germane to the failure or degradation than others. In other words, there are primary and secondary effects. This association provides a mechanism for ranking the effect. The implementing shall provide the EffectCorrelation property, but it is recommended that the implementation also provide the FailureRelationshipInitiated and Ranking properties.

If there are these cause and effect relationships, the RelatedElementCausingError association should be implemented to report the causes of the failure or degradation.

EXPERIMENTAL

26.1.10HealthState

The HealthState property in LogicalDevice defines the state for a particular component. The OperationalStatus defines operational status. For example, a disk or port may be taken off-line for service. The component's health may still be OK or not OK. The two properties, when used in combination, disambiguate the health of the component. For example, a OperationalStatus of 10 "Stopped" and a HealthState of 30 "Major Failure" means that the component is off-line and has failed. While a OperationalStatus of 10 "Stopped" and a HealthState of 5 "OK" for the very same component means that although the component is off-line, the component is still in good working order.

The HealthState of a component should not represent the health of any other component as well by way of a summary or aggregate health state. However, if the component is itself relies on other components for its health, because the component itself is an aggregate of components, then the HealthState may represent a summary HealthState by side-effect.

HealthState is a mandatory for all system device logical devices that are defined by the profile or subprofile that includes this package. It is recommended that HealthState is something other than 0 "Unknown". However, a component may report "Unknown" after it has reported one of the other HealthStates. When HealthState changes from 5 "OK", it is mandatory that a LogicalDevice report some other HealthState (e.g., 30 "Major Failure") before reporting 0 "Unknown". Such a requirement is necessary, so that the client can notice the adverse state change via polling or indication before the component is no longer responding.

26.2 Health and Fault Management Considerations

Not defined in this International Standard.

26.3 Cascading Considerations

Not defined in this International Standard.

26.4 Supported Subprofiles and Packages

Not defined in this International Standard.

26.5 Client Considerations and Recipes

Not defined in this International Standard.

26.6 Registered Name and Version

Health version 1.2.0 (Component Profile)

26.7 CIM Elements

26.7.1 General

Table 281 describes the CIM elements for Health.

Table 281 - CIM Elements for Health

Element Name	Requirement	Description
26.7.2 CIM_ComputerSystem	Mandatory	
26.7.3 CIM_LogicalDevice	Mandatory	
26.7.4 CIM_RelatedElementCausingError	Optional	

Table 281 - CIM Elements for Health

Element Name	Requirement	Description
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND SourceInstance.CIM_ComputerSystem::OperationalStatus[*] <> PreviousInstance.CIM_ComputerSystem::OperationalStatus[*]	Mandatory	CQL - Operational Status change of the device and application.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND SourceInstance.OperationalStatus <> PreviousInstance.OperationalStatus	Mandatory	Deprecated WQL - Operational Status change of the device and application.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_LogicalDevice AND SourceInstance.CIM_LogicalDevice::HealthState <> PreviousInstance.CIM_LogicalDevice::HealthState	Mandatory	CQL - Health State change of the logical component.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_LogicalDevice AND SourceInstance.HealthState <> PreviousInstance.HealthState	Mandatory	Deprecated WQL - Health State change of the logical component.

26.7.2 CIM_ComputerSystem

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 282 describes class CIM_ComputerSystem.

Table 282 - SMI Referenced Properties/Methods for CIM_ComputerSystem

Properties	Flags	Requirement	Description and Notes
CreationClassName		Mandatory	
Name		Mandatory	
OperationalStatus		Mandatory	Overall status of the Host.

26.7.3 CIM_LogicalDevice

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 283 describes class CIM_LogicalDevice.

Table 283 - SMI Referenced Properties/Methods for CIM_LogicalDevice

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	

Table 283 - SMI Referenced Properties/Methods for CIM_LogicalDevice

Properties	Flags	Requirement	Description and Notes
CreationClassName		Mandatory	
DeviceID		Mandatory	
HealthState		Mandatory	Reports the health of the component beyond the operational status.

26.7.4 CIM_RelatedElementCausingError

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 284 describes class CIM_RelatedElementCausingError.

Table 284 - SMI Referenced Properties/Methods for CIM_RelatedElementCausingError

Properties	Flags	Requirement	Description and Notes
FailureRelationshipInitiated		Optional	Reports the date and time when this cause and effect was created. The population of this property is RECOMMENDED.
EffectCorrelation		Mandatory	Describes the general nature of the cause and effect correlation.
Ranking		Optional	Describes the order of effect from 1, the highest effect, on. If there is only one of these associations between two elements, the ranking shall be 1. Once more associations are added, then it is RECOMMENDED that the implementation assist the client by stating which of the cause and effect relationship should be reviewed and addressed first. This property assists a client in accomplishing a triage of known problems.
Antecedent		Mandatory	Element causing the failure.
Dependent		Mandatory	

STABLE

STABLE

27 Job Control Subprofile

27.1 Description

27.1.1 General

In some profiles, some or all of the methods described may take some time to execute (longer than a HTTP time-out). In this case, a mechanism is needed to handle asynchronous execution of the method as a 'Job'.

This subprofile defines the constructs and behavior for job control for SNIA profiles that make use of the subprofile.

NOTE The subprofile describes a specific use of the constructs and properties involved. The actual CIM capability may be more, but this International Standard clearly states what clients may depend on in SNIA profiles that implement the Job Control Subprofile.

27.1.2 Instance Diagram

A normal instance diagram is provided in Figure 40.

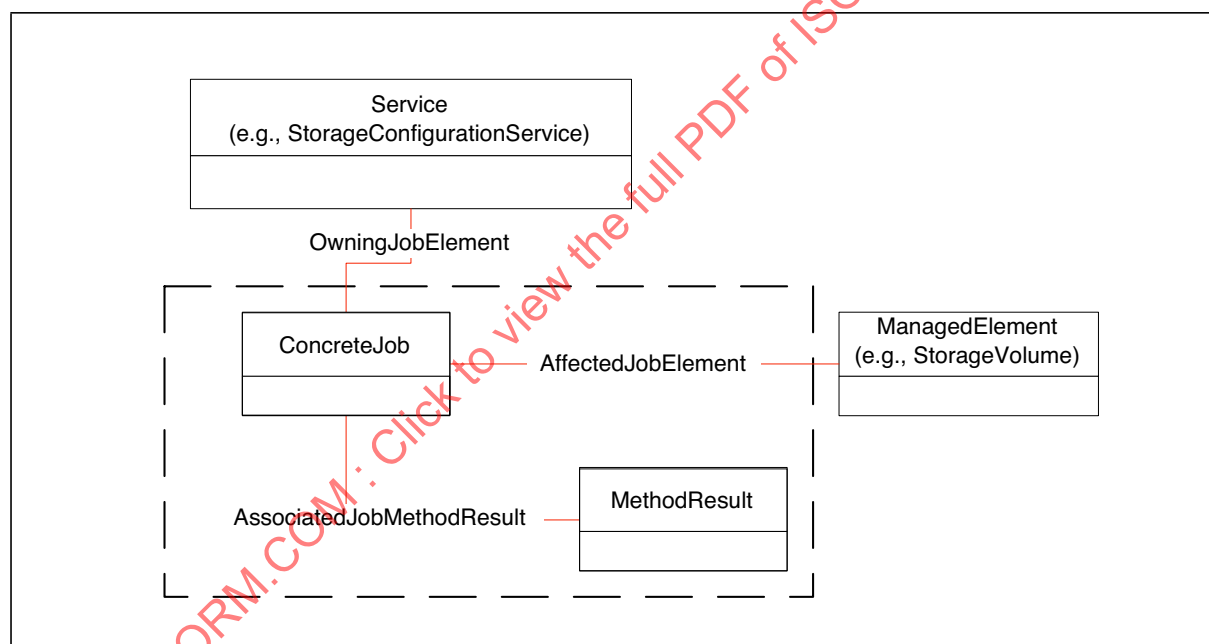


Figure 40 - Job Control Subprofile Model

When the Job Control Subprofile is implemented and a client executes a method that executes asynchronously, a reference to an instance of ConcreteJob is returned and the return value for the method is set to "Method parameters checked - job started".

The ConcreteJob instance allows the progress of the method to be checked, and instance Indications can be used to subscribe for Job completion.

The associations OwningJobElement and AffectedJobElement are used to indicate the service whose method created the job by side-effect and the element being affected by the job. The job itself may create, modify and/or delete many elements during its execution. The nature of this affect is the creation or deletion of the instances or associations or the modification of instance properties. These elements, albeit regular instances or associations, are said to be *affected* by the job. The elements linked by AffectedJobElement may change through the execution of the job, and in addition, the job may be associated to more than one Input and/or Output elements

or other elements affected by side-effect. Input and Output elements are those referenced by method parameters of the same type, input and output parameters respectively.

EXPERIMENTAL

The following set of rules defines the nature of the AffectedJobElement associations for a given job in terms of the references passed as parameters to the service method that spawned the job. Obviously, the distinction of Input element from Output element in the following rules only makes sense if these parameters are not both Input and Output elements.

- If all Elements created by the method exists immediately upon the return from the method, then AffectedJobElement shall reference the Output Element.
- If the Output Element, one or more, does not exist until the job has completed, the AffectedJobElement shall reference the Input Element until the job completes, at which time AffectedJobElement shall then reference the Output Element instead.
- In the event the job fails and the Output Element created during the job and referenced by AffectedJobElement is no longer available, AffectedJobElement shall revert to referencing the Input Element.
- If the method affects elements without referencing elements as Output parameters, then the AffectedJobElement Association shall reference the Input element, one or more.
- If the method only modifies the elements referenced with method parameters, then the AffectedJobElement association references the modified elements. Elements modified by the job shall be reference by this association.
- If the method affects elements but references no elements as either Input or Output parameters or the only Input elements referenced are those of the elements to be deleted, then AffectJobElement associations shall exist to other elements that are affected by the job.
- Other elements whose references are not used in the method invocation, but that are created or modified by side-effect of the job's execution shall be associated to the job via the AffectJobElement association, but may cease to be associated once the job has finished execution.

The lifetime of a completed job instance, and thus the AffectedJobElement association to the appropriate Element is currently implementation dependent. However, the set of AffectedJobElement associations to Input and Output element present when the job finishes execution shall remain until the job is deleted.

27.1.3 MethodResult

Jobs are produced by side effect of the invocation of an extrinsic method. Reporting the resulting Job is the purpose of this subprofile. The MethodResult class is used to report the extrinsic method called and the parameters passed to the method. In this way, third party observers of a CIMOM can tell what the job is and what it is doing. A MethodResult instance contains the LifeCycle indications that have been or would have been produced as the result of the extrinsic method invocation. That is, the instance contains the indications whether or not there were the appropriate indication subscription at the time the indication were produced.

A client may fetch the method lifecycle indication produced when the method was called from the PreCallIndication attribute. This indication, an instance of InstMethodCall, contains the input parameters provided by the client that called the method.

A client may fetch the method lifecycle indication produced once the method execution was completed from the PostCallIndication. This indication contains the input parameters provided by the client that called the method and output parameters returned by the method

implementation. Parameters that are both input and output parameters will contain the output parameter provided by the method implementation.

EXPERIMENTAL

27.1.4 OperationalStatus for Jobs

The OperationalStatus property is used to communicate that status of the job that is created. As such, it is critical that implementations are consistent in how this property is set. The values that shall be supported consistently are:

- 2 “OK” - combined with 17 “Completed” to indicate that the job completed with no error.
- 6 “Error” - combined with 17 “Completed” to indicate that the job did not complete normally and that an error occurred.
- 10 “Stopped” implies a clean and orderly stop.
- 17 “Completed” indicates the Job has completed its operation. This value should be combined with either 2 “OK” or 6 “Error”, so that a client can tell if the complete operation passed (Completed with OK), and failure (Completed with Error).

27.1.5 JobState for Jobs

The JobState property is used to communicate Job specific states and statuses.

- 2 “New” - Job was created but has not yet started.
- 3 “Starting” - Job has started.
- 4 “Running” - Job is current executing.
- 5 “Suspended” - Job has been suspended. The Job may be suspended for many reasons like it has been usurped by a higher priority or a client has suspended it (not described within this subprofile).
- 6 “Shutting Down” - Job is completing its work, has been terminated, or has been killed. The Job may be cleaning up after only having completed some of its work.
- 7 “Completed” - Job has completed normally, its work has been completed successfully.
- 8 “Terminated” - Job has been terminated
- 9 “Killed” - Job has been aborted. The Job may not cleanup after itself.
- 10 “Exception” - Job failed and is in some abnormal state. The client may fetch the error conditions from the job. See 27.5.2.

Table 285 maps the standard mapping between the OperationalStatus and JobState properties on ConcreteJob. The actual values of the properties are listed in Table 285 with the associated value from the property’s ValueMap qualifier.

Table 285 - OperationalStatus to Job State Mapping

OperationalStatus	JobState	Job is
2 “OK”, 17 “Completed”	7 “Completed”	Completed normally
6 “Error”, 17 “Completed”	10 “Exception”	Completed abnormally
10 “Stopped”	8 “Terminated”	Terminated
6 “Error”	9 “Killed”	Aborted / Killed
2 “OK”	4 “Running”	Executing

Table 285 - OperationalStatus to Job State Mapping

OperationalStatus	JobState	Job is
15 "Dormant"	2 "New"	Created but not yet executing
2 "OK", 8 "Starting"	3 "Starting"	Starting up
2 "OK"	5 "Suspended"	Suspended
2 "OK", 9 "Stopping"	6 "Shutting Down"	Terminated and potentially cleaning up
6 "Error"	6 "Shutting Down"	Killed and is aborting

27.1.6 Determining How Long a Job Remains after Execution

The Job shall report how long it will remain after it has finished executing, fails on its own, is terminated, or is killed. The TimeBeforeRemoval attribute reports a datetime offset.

The TimeBeforeRemoval and DeleteOnCompletion attributes are related. If the DeleteOnCompletion is FALSE, then the Job shall remain until it is explicitly deleted. If the DeleteOnCompletion is TRUE, then the Job shall exist for the length of time specified in the TimeBeforeRemoval attribute. An implementation may not support the setting of the DeleteOnCompletion attribute because it does not support the client modifying the Job instance.

The amount of time specified in the TimeBeforeRemoval should be five or more minutes. This amount of time allows a client to recognize that the Job has failed and retrieve the Error.

27.2 Health and Fault Management

The implementation should report CIM Errors from the ConcreteJob.GetError() method. See Clause 26 Health Package for details.

EXPERIMENTAL

The standards messages specific to this profile are listed in Table 286. See Clause 9 Standard Messages in *Information Technology – Storage Management, Part 2 Common Architecture* for a description of standard messages and the list all standard messages

Table 286 - Standard Message for Job Control Subprofile

Message ID	Message Name
DRM22	Job failed to start
DRM23	Job was halted

EXPERIMENTAL

27.3 Cascading Considerations

Not defined in this International Standard.

27.4 Support Subprofiles and Packages

Not defined in this International Standard.

27.5 Methods of the Profile

27.5.1 Job Modification

A Job instance may be modified. The DeleteOnCompletion and TimeBeforeRemoval properties are writable. If the intrinsic ModifyInstance method is supported, then the setting of both attributes shall be supported.

EXPERIMENTAL

27.5.2 Getting Error Conditions from Jobs

```
uint32 GetError(
    [Out, EmbeddedObject] string Error);
```

This method is used to fetch the reason for the job failure. The type of failure being report is when a Job stops executing on its own. That is, the Job was not killed or terminated. An Embedded Object, encoded in a string, shall returned if the method is both supported and the job has failed. The Job shall report the 10 "Exception" status when the Job has failed on its own.

The GetError method should be supported.

The Error string contains a Error instance. See Clause 26 Health Package for details on how to process this CIM Instance.

EXPERIMENTAL

27.5.3 Suspending, Killing or Terminating a Job

A Job may be suspended, terminated or killed. Suspending a Job means that the Job will not be executing and be suspended until it is resumed. Terminating a job means to request that the Job stop executing and that the Job clean-up its state prior to completing. Killing a job means to request that the Job abort executing, usually meaning there is little or no clean-up of Job state.

```
uint32 RequestStateChange(
    [In] RequestedState,
    [In] TimeoutPeriod);
```

A client may request a state change on the Job.

- RequestedState - The standard states that can be requested are "Start", "Suspend", "Terminate", "Kill", "Service". A new Job may be started. A suspended Job may be resumed, using the "Started" requested status. A executing Job may be suspended, terminated, or killed. A new or executing Job may be put into the "Service" state. The "Service" state is vendor specific. An implementation can indicate what state transitions are supported by not returning the 4 098 "Invalid State Transition" return code
- TimeoutPeriod - The client expects the state transition to occur within the specified amount of time. The implementation may support the method but not this parameter.

Return codes:

- 0 "Completed with No Error".
- 1 "Not Supported" - The method is not supported.
- 2 "Unknown/UnSpecified Error" - Failure for some vendor specific reason.
- 3 "Can not complete within Timeout Period" - The requested amount of time is less than how long the requested state transition takes.
- 4 "Failed".

- 5 “Invalid Parameters” - The parameters are incorrect.
- 6 “In Use” - Another client has requested a state change that has not completed.
- 4 096 “Method Parameters Checked - Transition Started” - The method can return before the state transition completes. This error code tells that calling that this situation has occurred.
- 4 097 “Invalid State Transition” - The state change requested is invalid for the current state. 4 098 “Use of Timeout Parameter Not Supported” - This implementation does not support the TimeoutPeriod parameter. A client may pass a NULL for the TimeoutPeriod and try again. There is no mechanism to determine what state changes are supported by a particular implementation. Such a mechanism is planned for a future version of this International Standard.
- 4 099 “Busy” - A state change is underway in the Job and, as such, the state can not be changed. An implementation may use this return code to indicate the job can not be suspended, killed, or terminated at all or in the current phase of execution.

27.6 Client Considerations and Recipes

If the operation will take a while (longer than an HTTP timeout), a handle to a newly minted ConcreteJob is returned. This allows the job to continue in the background. Note a few things:

- The job is associated to the Service via OwningJobElement and is also linked to the object being modified/created via AffectedJobElement. For example, a job to create a StorageVolume may start off pointing to a Pool until the Volume is instantiated at which point the association would change to the StorageVolume.
- These jobs do not have to get instantiated! If the method completes quickly, a null can be returned as a handle, as illustrated in Figure 41.
- It may take some time before the Job starts.
- A Job may be terminated or killed.
- Jobs may be modified.
- Jobs may be restarted.

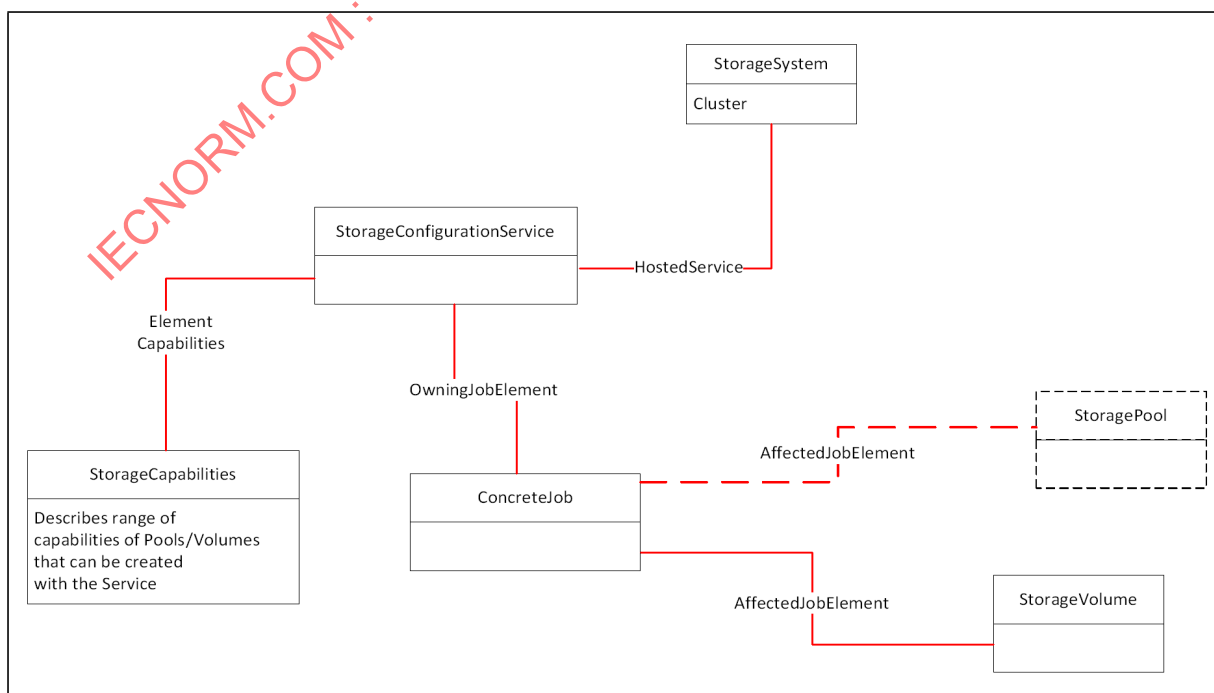


Figure 41 - Storage Configuration

27.7 Registered Name and Version

Job Control version 1.5.0 (Component Profile).

27.8 CIM Elements**27.8.1 General**

Table 287 describes the CIM elements for Job Control.

Table 287 - CIM Elements for Job Control

Element Name	Requirement	Description
27.8.2 CIM_AffectedJobElement	Mandatory	
27.8.3 CIM_AssociatedJobMethodResult	Mandatory	
27.8.4 CIM_ConcreteJob	Mandatory	
27.8.5 CIM_MethodResult	Mandatory	
27.8.6 CIM_OwningJobElement	Mandatory	
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ConcreteJob AND SourceInstance.CIM_ConcreteJob::JobStatus <> PreviousInstance.CIM_ConcreteJob::JobStatus	Optional	CQL -Deprecated. Modification of Job Status for a Concrete Job.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ConcreteJob AND SourceInstance.JobStatus <> PreviousInstance.JobStatus	Optional	Deprecated WQL -Deprecated: Modification of Job Status for a Concrete Job.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ConcreteJob AND SourceInstance.CIM_ConcreteJob::PercentComplete <> PreviousInstance.CIM_ConcreteJob::PercentComplete	Mandatory	CQL -Modification of Percentage Complete for a Concrete Job.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ConcreteJob AND SourceInstance.PercentComplete <> PreviousInstance.PercentComplete	Mandatory	Deprecated WQL -Modification of Percentage Complete for a Concrete Job.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ConcreteJob AND ANY SourceInstance.CIM_ConcreteJob::OperationalStatus[*] = 17 AND ANY SourceInstance.CIM_ConcreteJob::OperationalStatus[*] = 2	Mandatory	CQL -Modification of Operational Status for a Concrete Job to 'Complete' and 'OK'.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ConcreteJob AND SourceInstance.OperationalStatus = 17 AND SourceInstance.OperationalStatus = 2	Mandatory	Deprecated WQL -Modification of Operational Status for a Concrete Job to 'Complete' and 'OK'.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ConcreteJob AND ANY SourceInstance.CIM_ConcreteJob::OperationalStatus[*] = 17 AND ANY SourceInstance.CIM_ConcreteJob::OperationalStatus[*] = 6	Mandatory	CQL -Modification of Operational Status for a Concrete Job to 'Complete' and 'Error'.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ConcreteJob AND SourceInstance.OperationalStatus = 17 AND SourceInstance.OperationalStatus = 6	Mandatory	Deprecated WQL -Modification of Operational Status for a Concrete Job to 'Complete' and 'Error'.

Table 287 - CIM Elements for Job Control

Element Name	Requirement	Description
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ConcreteJob AND SourceInstance.CIM_ConcreteJob::JobState <> PreviousInstance.CIM_ConcreteJob::JobState	Mandatory	CQL -Modification of Job State for a Concrete Job.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ConcreteJob AND SourceInstance.JobState <> PreviousInstance.JobState	Mandatory	Deprecated WQL -Modification of Job State for a Concrete Job.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_ConcreteJob	Mandatory	Creation of a ConcreteJob.

27.8.2 CIM_AffectedJobElement

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 288 describes class CIM_AffectedJobElement.

Table 288 - SMI Referenced Properties/Methods for CIM_AffectedJobElement

Properties	Flags	Requirement	Description and Notes
AffectedElement		Mandatory	The ManagedElement affected by the execution of the Job.
AffectingElement		Mandatory	The Job that is affecting the ManagedElement.

27.8.3 CIM_AssociatedJobMethodResult

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 289 describes class CIM_AssociatedJobMethodResult.

Table 289 - SMI Referenced Properties/Methods for CIM_AssociatedJobMethodResult

Properties	Flags	Requirement	Description and Notes
Job		Mandatory	The Job that has parameters.
JobParameters		Mandatory	The parameters for the method which by side-effect created the Job.

27.8.4 CIM_ConcreteJob

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 290 describes class CIM_ConcreteJob.

Table 290 - SMI Referenced Properties/Methods for CIM_ConcreteJob

Properties	Flags	Requirement	Description and Notes
InstanceID		Mandatory	
Name		Mandatory	The user-friendly name for this instance of Job. In addition, the user-friendly name can be used as a property for a search or query. (Note that Name does not have to be unique within a namespace.)
OperationalStatus		Mandatory	Describes whether the Job is running or not.
JobStatus		Optional	Add additional detail beyond OperationalStatus about the runtime status of the Job. This property is vendor specific.
JobState		Mandatory	Add additional detail beyond the OperationalStatus about the runtime state of the Job.
ElapsedTime		Optional	The time interval that the Job has been executing or the total execution time if the Job is complete.
PercentComplete		Mandatory	The percentage of the job that has completed at the time that this value is requested. Optimally, the percentage should reflect the amount of work accomplished in relation to the amount of work left to be done. 0 percent complete means that the job has not started and 100 percent complete means the job has finished all its work. However, in the degenerate case, 50 percent complete means that the job is running and may remain that way until the job completes.
DeleteOnCompletion		Mandatory	Indicates whether or not the job should be automatically deleted upon completion. If this property is set to false and the job completes, then the extrinsic method DeleteInstance shall be used to delete the job versus updating this property. Even if the Job is set to delete on completion, the job shall remain for some period of time, see GetError() method.
ErrorCode		Optional	A vendor specific error code. This is set to zero if the job completed without error.
ErrorDescription		Optional	A free form string containing the vendor error description.

Table 290 - SMI Referenced Properties/Methods for CIM_ConcreteJob

Properties	Flags	Requirement	Description and Notes
TimeBeforeRemoval		Mandatory	The amount of time the job will exist after the execution of the Job if DeleteOnCompletion is set to FALSE. Jobs that complete successfully or fail shall remaining for at least this period of time before being removed from the model (CIMOM).
GetError()		Mandatory	This method is used to retrieve the error that caused the Job to fail. The Job shall remain in the model long enough to allow a client to a) notice that the job was stopped executing and b) to retrieve the error using this method. There are no requirements for how long the job shall remain. However, it is suggested that the Job remain for at least five minutes. JobStatus=10 (Exception) tell the client that the job failed and this method can be called to retrieve the reason why embedded in the CIM_Error, see GetError() method.
RequestStateChange()		Optional	This method changes the state of the job. The client may suspend, terminate, or shutdown the job. To terminate a job means to request a clean shutdown of the job, have it finish some portion of its work and terminate or to roll back the changes done by the job to date. The implementation can choose the behavior. To kill a job means to abort the job, perhaps leaving some element of the work partially done and in an unknown state.

27.8.5 CIM_MethodResult

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 291 describes class CIM_MethodResult.

Table 291 - SMI Referenced Properties/Methods for CIM_MethodResult

Properties	Flags	Requirement	Description and Notes
InstanceID		Mandatory	
PreCallIndication		Mandatory	Contains a copy of the CIM_InstMethodCall produced when the configuration or control change method was called. This Embedded Instance shall contain the configuration or control change extrinsic method name (MethodName) and parameters (MethodParameters).
PostCallIndication		Mandatory	Contains a copy of the CIM_InstMethodCall produced when the configuration or control change method has completed execution and control was returned to the client. This Embedded Instance shall contain the configuration or control change extrinsic method name (MethodName) and parameters (MethodParameters).

27.8.6 CIM_OwningJobElement

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 292 describes class CIM_OwningJobElement.

Table 292 - SMI Referenced Properties/Methods for CIM_OwningJobElement

Properties	Flags	Requirement	Description and Notes
OwningElement		Mandatory	The ManagedElement responsible for the creation of the Job. (e.g., StorageConfigurationService).
OwnedElement		Mandatory	The Job created by the ManagedElement.

STABLE

STABLE

28 Location Subprofile

28.1 Description

28.1.1 General

Associated with product information, a PhysicalPackage may also have a location. This is indicated using an instance of a Location class and the PhysicalElementLocation association.

28.1.2 Instance Diagram

Figure 42 illustrates a typical instance diagram.

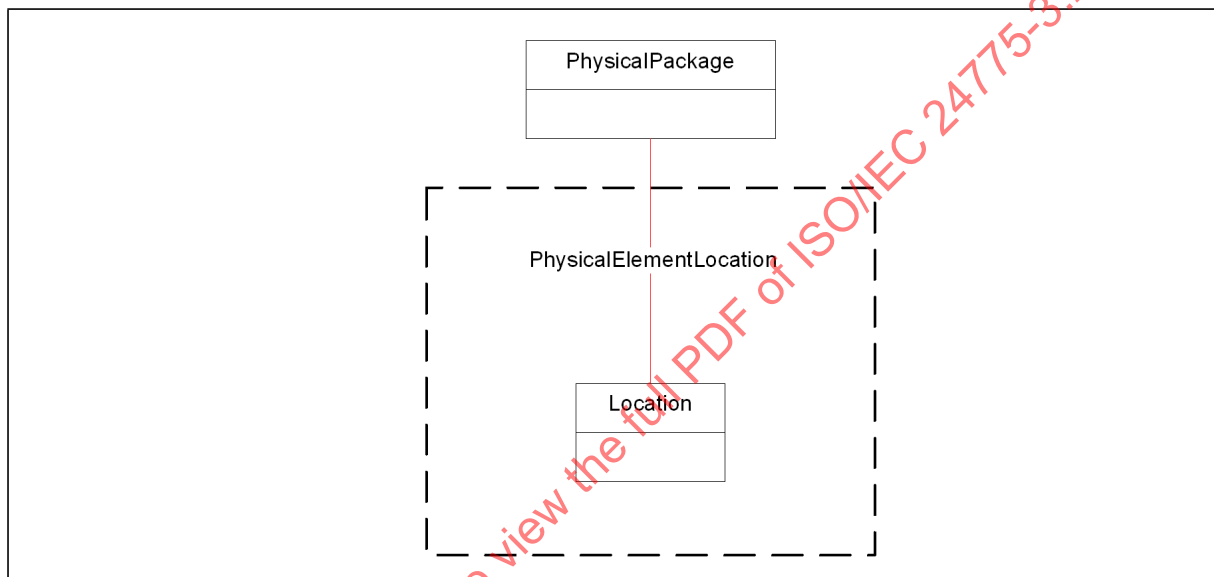


Figure 42 - Location Instance

28.2 Health and Fault Management Considerations

Not defined in this International Standard.

28.3 Cascading Considerations

Not defined in this International Standard.

28.4 Supported Subprofiles and Packages

None.

28.5 Methods of the Profile

None.

28.6 Client Considerations and Recipes

None.

28.7 Registered Name and Version

Location version 1.4.0 (Component Profile).

28.8 CIM Elements

28.8.1 General

Table 293 describes the CIM elements for Location.

Table 293 - CIM Elements for Location

Element Name	Requirement	Description
28.8.2 CIM_Location	Mandatory	
28.8.3 CIM_PhysicalElementLocation	Mandatory	Associates the location to package.

28.8.2 CIM_Location

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 294 describes class CIM_Location.

Table 294 - SMI Referenced Properties/Methods for CIM_Location

Properties	Flags	Requirement	Description and Notes
Name		Mandatory	A free-form string defining a label for the Location.
PhysicalPosition		Mandatory	A free-form string indicating the placement of a PhysicalElement.
ElementName		Optional	User-friendly name.
Address		Optional	A free-form string indicating a street, building or other type of address for the PhysicalElement's Location.

28.8.3 CIM_PhysicalElementLocation

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 295 describes class CIM_PhysicalElementLocation.

Table 295 - SMI Referenced Properties/Methods for CIM_PhysicalElementLocation

Properties	Flags	Requirement	Description and Notes
Element		Mandatory	
PhysicalLocation		Mandatory	

STABLE

DEPRECATED

29 Extra Capacity Set Subprofile

The functionality of the Extra Capacity Set Subprofile has been replaced by the 31 Multiple Computer System Subprofile.

The Extra Capacity Set Subprofile is defined in section B.8 of SMI-S 1.0.2.

DEPRECATED

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

DEPRECATED

30 Cluster Subprofile

The functionality of the Cluster Subprofile has been subsumed by 31 Multiple Computer System Subprofile.

The Cluster Subprofile is defined in 7.3.3.3 of SMI-S 1.0.2. Any instrumentation that complies to the Multiple Computer System Subprofile defined in this International Standard may also claim compliance to that version of the Cluster Subprofile and may register as both a 1.1.0 Multiple Computer System Subprofile and 1.0.2 Cluster Subprofile.

DEPRECATED

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

STABLE

31 Multiple Computer System Subprofile

31.1 Description

31.1.1 General

The Multiple Computer System Subprofile models multiple systems that cooperate to present a “virtual” computer system with additional capabilities or redundancy. This virtual aggregate system is sometimes referred to as a cluster and is illustrated in Figure 43.

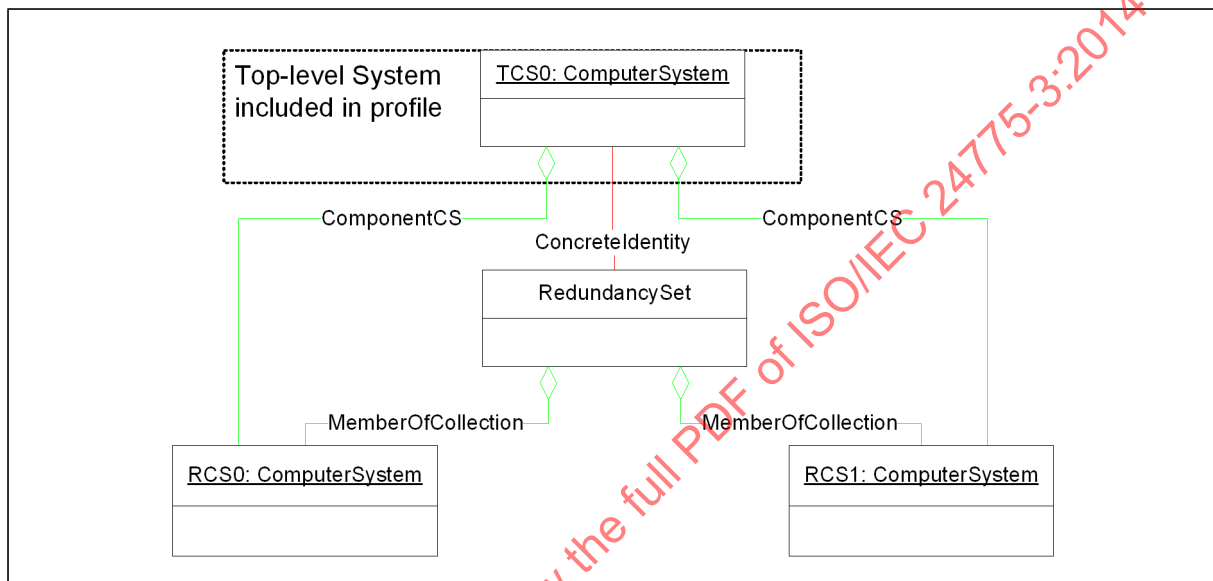


Figure 43 - Two Redundant Systems Instance Diagram

The general pattern for the redundancy aspect of Multiple Systems uses an instance of RedundancySet to aggregate multiple “real” ComputerSystem instances (labeled RCS0 and RCS1 in the diagram). Another ComputerSystem instance (TCS0) is associated to the RedundancySet instance using a ConcreteIdentity association and is associated to the real ComputerSystems using ComponentCS.

31.1.2 Top Level System

The top (“virtual”) system in this diagram (labeled TCS0) is referred to as the Top Level System. Note that for single-system configurations, the top-level system is the only system. Top-level systems have characteristics different from the underlying ComputerSystem instances.

The Top Level System is associated to the registered profile described in Clause 41 Server Profile. Other elements such as LogicalDevices (ports, volumes), ServiceAccessPoints, and Services are associated to the top-level system if these elements are supported by multiple underlying systems (for example, the underlying systems provide failover and/or load balancing). Alternatively, elements can be associated to an underlying system if that system is a single point of failure. For example, a RAID array may associate StorageVolume instances to a top-level system since these are available when one underlying system (RAID controller) fails, all the port elements are associated to one underlying system because the ports become unavailable when this system fails.

The Dedicated property is required for top-level systems. Each profile defines the values that are appropriate for Dedicated.

31.1.3 Non-Top-Level Systems

Each ComputerSystem instance shall have a unique Name property. For non-top-level systems, Name may be vendor-unique; in which case, NameFormat shall be set to "Other".

ComputerSystem.Dedicated should not be used in non-top-level systems.

Non-top-level systems shall not be associated to registered profiles or subprofiles.

Each non-top-level ComputerSystem shall be associated to the top-level system using ComponentCS. Note that non-top-level systems may not be members of a RedundancySet. For example, a top-level system may be associated to a RedundancySet with two systems as described in Figure 43 and also associated via ComponentCS to another Computer (not a member of a RedundancySet) representing a service processor.

31.1.4 Types of RedundancySets

The TypeOfSet property of RedundancySet is a list describing the types of redundancy. Its values are summarized in Table 296.

Table 296 - Redundancy Type

Redundancy Type	Description
N+1	All ComputerSystems are active, are unaware and function independently of one another. However, there exists at least one extra ComputerSystem to achieve functionality.
Load Balanced	All computer systems are active. However, their functionality is not independent of each other. Their functioning is determined by some sort of load balancing algorithm (implemented in hardware and/or software). 'Sparing' is implied (i.e., each computer system can be a spare for the other(s)).
Sparing	All computer systems are active and are aware of each other. However, their functionality is independent until failover. Each computer system can be a spare for the other(s).
Limited Sparing	All members are active, and they may or may not be aware of each other and they are not spares for each other. Instead, their redundancy is indicated by the IsSpare relationship.
Other/Unspecified	The relationship between the computer systems is not specified.

31.1.5 Multiple Tiers of Systems

The diagram above describes two tiers of systems; the real systems (labeled RCS0 and RCS1) in the lower tier are aggregated into a top-level system (TCS0) in the upper tier. There may be more than two tiers, as depicted in Figure 44.

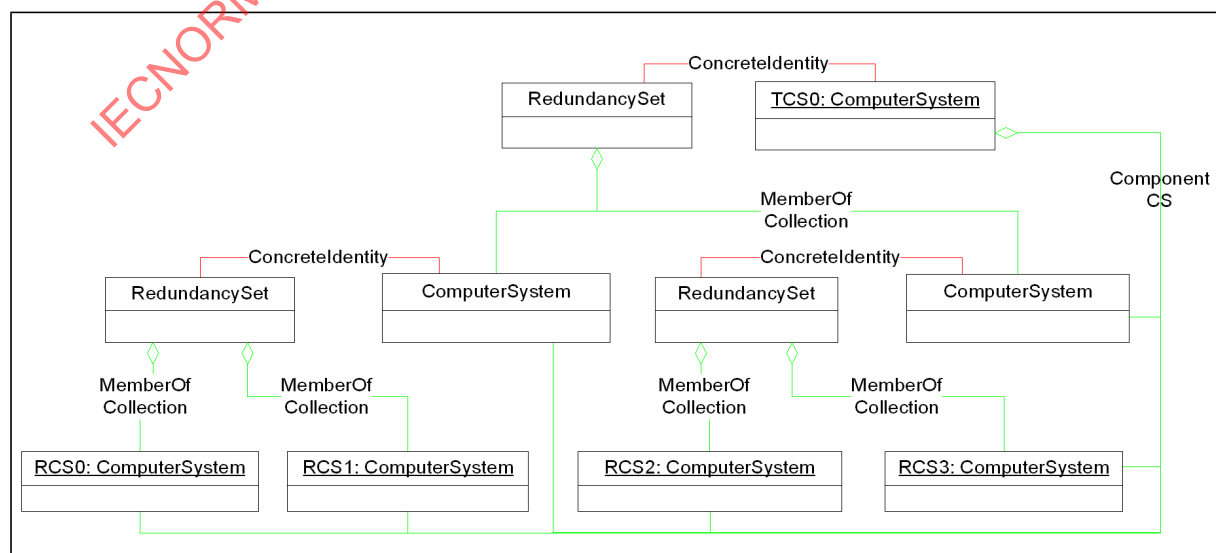


Figure 44 - Multiple Redundancy Tier Instance Diagram

The systems in the bottom tier (RCS0-RCS3) represent "real" systems.

RedundancySet.TypeOfSet can be used as part of multiple tier configurations to describe different types of redundancy at different tiers. For example, a virtualization system has four controllers that operate in pairwise redundancy. This could be modeled using the model in the diagram above and setting TypeOfSet in the top RedundancySet to "N+1" and setting TypeOfSet to "LoadBalancing" in the lower two RedundancySets.

31.1.6 Associations between ComputerSystems and other Logical Elements

SystemDevice associates device (subclasses of LogicalDevice such as LogicalPort or StorageVolume) and ComputerSystem instances. The cardinality of SystemDevice is one-to-many. A LogicalDevice may be associated with one and only one ComputerSystem. If the device availability is equivalent to that of the top-level system, it shall be associated to the top-level system via SystemDevice. If the device may become unavailable while the system as a whole remains available, the device shall be associated to a non-top-level system that has availability equivalent to the device. This system could be a real system or a system in an intermediate tier (representing some redundancy less than full redundancy).

This same approach shall be used for all other logical CIM elements with associations to systems. For example, HostedService and HostedAccessPoint shall associate elements (services, access points, and protocol endpoints) to the ComputerSystem with availability to the element.

Based on the arrangement of systems in figure 31, associations from systems to service and capabilities classes shall not be lower than associations to other classes. For the purpose of formally stating this rule, each ComputerSystem is assigned a level number. The profile's top-level ComputerSystem has level number 0. The ComputerSystem instances that are members of RedundancySets associated via ConcretelDentity to the top-level system have level number 1. The members of redundancy sets associated to the level number 1 systems via ConcretelDentity have level number 2. In general, the ComputerSystem members of redundancy sets associated to the level number n systems via ConcretelDentity have level number $n + 1$. The level of non-system objects is the level of the ComputerSystem instance associated to the object via associations such as SystemDevice, HostedAccessPoint, HostedService, or ElementCapabilities.

Figure 45 demonstrates these system level numbers using the same configuration from Figure 44. Note that ComponentCS diagrams are omitted from this diagram to avoid clutter.

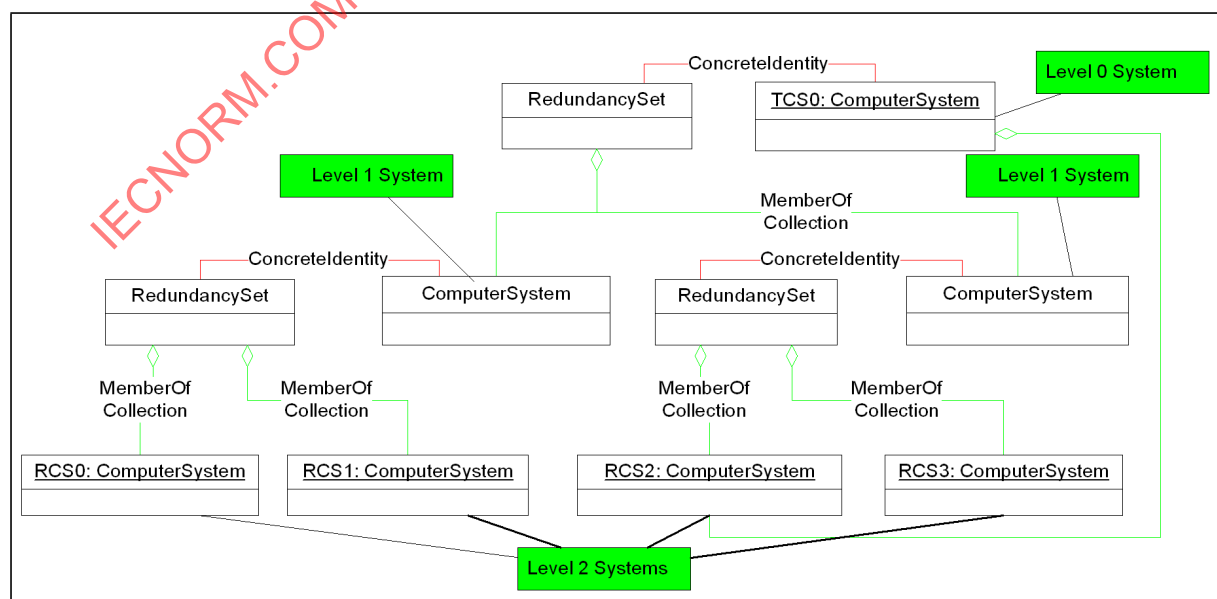


Figure 45 - System Level Numbers

All subclasses of `CIM_Service` and `CIM_Capabilities` shall have a level number less than or equal to the level number of storage classes (ports, volumes, etc.) that are influenced by the properties and methods of the `Service` and `Capabilities` classes. In some cases, different storage classes are influenced by different `Service` or `Capabilities` classes. The “level number less than or equal to” requirement may apply differently to different `Service/Capabilities` classes. It is always valid to associate `Service` and `Capabilities` classes to the top-level `ComputerSystem` since the level number of the top-level system (0) is always less than or equal to the level number of any other system.

Example 1 - An array with two controllers is modeled as a top-level `ComputerSystem` with real systems representing the controllers. The system's storage volumes remain available when one controller fails, but each `LogicalPort` becomes unavailable when a controller fails. The `StorageVolumes` should be associated to the top-level `ComputerSystem` and the `LogicalPorts` should be associated to one of the real `ComputerSystems`.

Example 2 - An array with four pair-wise redundant controllers. Each `LogicalPort` is associated with a pair of controllers - if one controller in a pair fails, the port is still accessible through the alternate controller. This corresponds to Figure 44. The ports should be associated with one of the `ComputerSystems` in the middle tier.

A provider shall delete and create associations between `ComputerSystems` and logical elements (e.g., ports, logical devices) during failover or fallback to represent changes in availability. This includes `SystemDevice`, `HostedAccessPoint`, `HostedService`, or `HostedFileSystem` associations (and other associations weak to systems). The effect of the creation and deletion of associations is to switch these elements from one `ComputerSystem` to another. The profiles that include Multiple Computer System Subprofile shall specify the affected associations and indications for creation and deletion of these associations.

31.1.7 Associations between `ComputerSystems` and `PhysicalPackages` and `Products`

The relationship between `ComputerSystems`, `PhysicalPackages`, and `Products` is defined in the Physical Package Package (see 32 Physical Package Package) which may be required by the profile including the Multiple Computer System Subprofile. Typically, the top-level system is associated to a `PhysicalPackage` which is associated to a `Product`. Non-top-level systems may also be associated to `PhysicalPackage` and indirectly to a `Product`. If all underlying `ComputerSystems` share the same physical package, a single `PhysicalPackage` should be associated to the upper `ComputerSystem`.

The relationships between `ComputerSystems`, redundancy sets, and CIM logical elements serve as a redundancy topology - informing the client of the availability of subsets of logical elements. The relationships between `PhysicalPackages` and logical elements serve as a physical topology. These two topologies need not be equivalent. Consider these examples:

Example 1: a RAID array with a single controller (no redundancy); the controller and all backend disks are housed in a single chassis. This is modeled as a single `ComputerSystem`, no `RedundancySets`, no `ComponentCS` associations, and a single `PhysicalPackage` with a single associated `Product`.

Example 2: a RAID array with two redundant controllers; both controllers and all backend disks share a single chassis. In this case, the redundancy topology matches Figure 43. The top-level `ComputerSystem` is associated to a `PhysicalPackage` with a single associated `Product`.

Example 3: two arrays described in example 1 are assembled as part of common rack and sold as a single product. Note that although there are two controllers, there is no redundancy - the two controllers act completely independently. This is modeled as two top-level computer systems attached to separate `PhysicalPackages` (representing the two internal chassis). These two `PhysicalPackages` have a `Container` association to third `PhysicalPackage` representing the assembly - which has an association to a `Product`.

Example 4: two arrays described in Example 1 are assembled as part of a common rack and also share a high-speed trunk and a mutual failover capability. This failover capability means the two controllers share a `RedundancySet` and common top-level system. The result is similar

to example 2, but each real ComputerSystem is now associated to separate PhysicalPackages which have Container associations to a common PhysicalPackage.

31.1.8 Storage Systems without Multiple Systems

In configurations where the instrumentation does not model multiple ComputerSystem instances, all the associations described above reference the one and only ComputerSystem.

31.1.9 Durable Names and Correlatable IDs of the Subprofile

This subprofile does not impose any requirements on names. The requirements for ComputerSystem names are defined in the profiles that depend on Multiple Computer System Subprofile and in *Information Technology – Storage Management, Part 2 Common Architecture* 8 Correlatable and Durable Names. Clients should not expect that a network name or IP address is exposed as a ComputerSystem property. The Access Points Subprofile should be used to model a network access point.

31.2 Health and Fault Management Considerations

The requirements for OperationalStatus of a ComputerSystem are discussed in Clause 26 Health Package.

31.3 Cascading Considerations

None.

31.4 Supported Subprofiles and Packages

Table 297 describes the supported profiles for Multiple Computer System.

Table 297 - Supported Profiles for Multiple Computer System

Profile Name	Organization	Version	Requirement	Description
Storage Server Asymmetry	SNIA	1.4.0	Optional	

31.5 Methods of the Profile

This subprofile does not include any extrinsic methods. A client may use this subprofile to discover information about the topology of computer systems, but cannot change the topology.

31.6 Client Considerations and Recipes

31.6.1 General

A client cannot generally, interoperably navigate the redundancy topology using ComponentCS because some Component CS associations may not parallel RedundancySet associations. But a client may use ComponentCS selectively to speed up certain tasks. In particular, a client may locate the top-level system from other ComputerSystems using ComponentCS.

31.6.2 Find Top-level Computer Systems

Top-level systems are the only objects in SMI-S associated to RegisteredProfile via ElementConformsToProfile. (See 42.5.5.)

31.6.3 Find the Top-level Computer System for any LogicalDevice

```
/
// DESCRIPTION:
// Find the Top-level Computer System for any CIM_LogicalDevice
//
// Preconditions:
// $Device - Reference the LogicalDevice
//
```

```

// Find Systems associated to $Device
$Systems->[] = AssociatorNames($Device->, // ObjectName
    "CIM_SystemDevice", // AssocClass
    "CIM_System", // ResultClass
    "PartComponent", // Role
    "GroupComponent") // ResultRole
if ($Systems == null || $Systems->[].size != 1) {
    <ERROR! must be exactly one ComputerSystem Associated via
        SystemDevice to each LogicalDevice instance>
}

// System->[0] is the associated system; see if it's the
// top-level system for the scoping profile. All ComponentCS
// association GroupComponent references must refer to the
// profile's top-level system.
$UpperSystems->[] = AssociatorNames($System->[0],
    "CIM_ComponentCS", // AssocClass
    "CIM_ComputerSystem", // ResultClass
    "PartComponent", // Role
    "GroupComponent") // ResultRole
if ($UpperSystems != null && $UpperSystems->[].size > 1) {
    // The restriction below is a characteristic of this subprofile
    // and matches the DMTF Partition white paper.
    <ERROR! must be no more than one ComputerSystem Associated
        via ComponentCS to each LogicalDevice instance>
}
// If an upper system was found, it must be the top-level
// system; if not, then the system associated to the device
// must be the top-level system
if ($UpperSystems->[].size == 1) {
    $TopLevelSystem = $UpperSystems->[0]
} else {
    $TopLevelSystem = $System->[0]
}

// The remaining steps are not needed to locate the top-level
// system, but validate the classes and associations.
//
// The system associated to the device may also be part of a
// RedundancySet.
// If so, follow a chain from that system to the RedundancySet, then
// follow ConcreteIdentity to a system - then check to see if it has
// ComponentCS to the top-level system. Keep iterating till no more
// RedundancySets - this must be the same system as TopLevelSystem.
do {
    // Get the RedundancySet that $System->[0] is a member of
    $RedundancySets->[] = AssociatorNames($System->[0],
        "CIM_MemberOfCollection",
        "CIM_RedundancySet",
        "Member",
        "Collection")
}

```

```
if ($RedundancySets == null || $RedundancySets->[].size == 0) {
    #InARedundancySet = false
} else {
    #InARedundancySet = true
    // Error is more than one RedundancySet
    if ($RedundancySets->[].size != 1) {
        <ERROR: A system cannot be the member of multiple
            RedundancySets>
    }
    $Systems->[] = AssociatorNames($RedundancySets->[0], //
        ObjectName
        "CIM_LogicalIdentity", // AssocClass
        "CIM_System", // ResultClass
        "SameElement", // Role
        "SystemElement") // ResultRole
    if ($Systems == null || $Systems->[].size != 1) {
        <ERROR: There must be exactly one System associated to each
            RedundancySet>
    }
    // if System->[0] is not the TopLevelSystem, it must have
        ComponentCS
    if ($System->[0] != $TopLevelSystem) {
        $UpperSystems->[] = AssociatorNames($System->[0],
            "CIM_ComponentCS", // AssocClass
            "CIM_ComputerSystem", // ResultClass
            "PartComponent", // Role
            "GroupComponent") // ResultRole
        if ($UpperSystems == null && $UpperSystems->[].size != 1) {
            <ERROR: must be no more than one ComputerSystem Associated
                via ComponentCS to each LogicalDevice instance>
        }
        if ($UpperSystems->[0] != $TopLevelSystem) {
            <ERROR: The one end of every ComponentCS must be the Top
                Level
                system>
        }
    }
}
} while (#InARedundancySet)

// The top-level system must be associated to a RegisteredProfile
$Profiles->[] = AssociatorNames($TopLevelSystem,
    "CIM_ElementConformsToProfile",
    "CIM_RegisteredProfile",
    NULL, NULL)
if ($Profiles == null || $Profiles->[].size == 0) {
    <ERROR: Top-Level system not associated to RegisteredProfile>
}
```

31.7 Registered Name and Version

Multiple Computer System version 1.2.0 (Component Profile).

31.8 CIM Elements

31.8.1 General

Table 298 describes the CIM elements for Multiple Computer System.

Table 298 - CIM Elements for Multiple Computer System

Element Name	Requirement	Description
31.8.2 CIM_ComponentCS	Mandatory	Associates non-top-level systems to the top-level system.
31.8.3 CIM_ComputerSystem (Non-Top-Level System)	Mandatory	Non-Top-level System.
31.8.4 CIM_ConcreteIdentity	Mandatory	Associates aggregate (possibly top-level) ComputerSystem and RedundancySet.
31.8.5 CIM_IsSpare	Optional	Associates the ComputerSystem that may be used as a spare to the RedundancySet of ActiveComputerSystem.
31.8.6 CIM_MemberOfCollection	Mandatory	Associates RedundancySet and its member ComputerSystems.
31.8.7 CIM_RedundancySet	Mandatory	
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_ComputerSystem	Mandatory	Creation of a ComputerSystem instance.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_ComputerSystem	Mandatory	Deletion of a ComputerSystem instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND SourceInstance.OperationalStatus <> PreviousInstance.OperationalStatus	Mandatory	Deprecated WQL - Change of Operational Status of a ComputerSystem instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND SourceInstance.CIM_ComputerSystem::OperationalStatus <> PreviousInstance.CIM_ComputerSystem::OperationalStatus	Mandatory	CQL - Change of Operational Status of a ComputerSystem instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_RedundancySet AND SourceInstance.RedundancyStatus <> PreviousInstance.RedundancyStatus	Mandatory	Deprecated WQL - Change of redundancy status.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_RedundancySet AND SourceInstance.CIM_RedundancySet::RedundancyStatus <> PreviousInstance.CIM_RedundancySet::RedundancyStatus	Mandatory	CQL - Change of redundancy status.

31.8.2 CIM_ComponentCS

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 299 describes class CIM_ComponentCS.

Table 299 - SMI Referenced Properties/Methods for CIM_ComponentCS

Properties	Flags	Requirement	Description and Notes
GroupComponent		Mandatory	The Top-Level ComputerSystem; shall be associated to a RegisteredProfile.
PartComponent		Mandatory	The contained (Sub)ComputerSystem.

31.8.3 CIM_ComputerSystem (Non-Top-Level System)

Non-Top-level system.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 300 describes class CIM_ComputerSystem (Non-Top-Level System).

Table 300 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Non-Top-Level System)

Properties	Flags	Requirement	Description and Notes
CreationClassName		Mandatory	
Name		Mandatory	
NameFormat		Mandatory	Non-top-level system names are not correlatable, any format is valid.
ElementName		Mandatory	
OperationalStatus		Mandatory	

31.8.4 CIM_ConcreteIdentity

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 301 describes class CIM_ConcreteIdentity.

Table 301 - SMI Referenced Properties/Methods for CIM_ConcreteIdentity

Properties	Flags	Requirement	Description and Notes
SystemElement		Mandatory	
SameElement		Mandatory	

31.8.5 CIM_IsSpare

Associates the ComputerSystem that may be used as a spare to the RedundancySet of ActiveComputerSystem.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 302 describes class CIM_IsSpare.

Table 302 - SMI Referenced Properties/Methods for CIM_IsSpare

Properties	Flags	Requirement	Description and Notes
SpareStatus		Mandatory	
FailoverSupported		Mandatory	
Dependent		Mandatory	The RedundancySet.
Antecedent		Mandatory	The spare system.

31.8.6 CIM_MemberOfCollection

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 303 describes class CIM_MemberOfCollection.

Table 303 - SMI Referenced Properties/Methods for CIM_MemberOfCollection

Properties	Flags	Requirement	Description and Notes
Collection		Mandatory	
Member		Mandatory	

31.8.7 CIM_RedundancySet

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 304 describes class CIM_RedundancySet.

Table 304 - SMI Referenced Properties/Methods for CIM_RedundancySet

Properties	Flags	Requirement	Description and Notes
InstanceID		Mandatory	
ElementName		Mandatory	
RedundancyStatus		Mandatory	The redundancy status shall be either 'Unknown' 0, 'Redundant' 2, or 'Redundancy Lost'. The implementation should report 2 or 3 most of the time, although it may report 0 sometimes. It should report 2 when there is at least one spare per the RedundancySet. It should report 3 when there are no more spares (via IsSpare association) per the RedundancySet.
TypeOfSet		Mandatory	

STABLE

STABLE

32 Physical Package Package

32.1 Description

32.1.1 General

Physical Package Package models information about a storage system's physical package and optionally about internal sub-packages. A System is 'realized' using a SystemPackaging association to a PhysicalPackage (or a subclasses such as Chassis). The physical containment model can then be built up using Container associations and subclasses (such as PackageInChassis).

Physical elements are described as products using the Product class and ProductPhysicalComponent associations, as shown in Figure 46. The Product instances may be built up into a hierarchy using the ProductParentChild association. The Product class holds information such as vendor name, serial number and version.

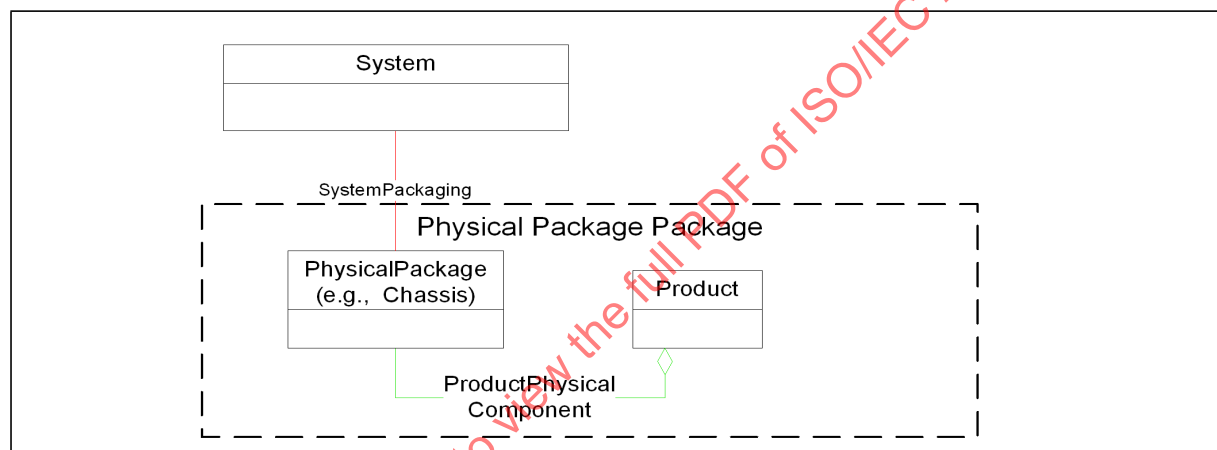


Figure 46 - Physical Package Package Mandatory Classes

32.1.2 Well Defined Subcomponents

In addition to defining physical packages at the "System" level, PhysicalPackage may also be defined at a lower, subcomponent level. For example, PhysicalPackage is used in the Disk Drive Lite Subprofile and for devices supported by storage media libraries (e.g., TapeDrive and ChangerDevice). If the subcomponents are supported by the Profile, they should model their physical packaging. When subcomponents are modeled, there shall be a container relationship between their physical package and the containing package (e.g., the System level physical package). In addition, there shall be a ProductParentChild association between the subcomponent Product and the parent Product.

The Physical Package constructs may also be used to model other aspects of the environment. However, this is not mandatory. Note that each logical device may be realized by a card. The cards are contained in a controller chassis.

When establishing physical packages for logical device subcomponents (e.g., disk drives, changers, etc.) the provider shall populate both Container and Realizes associations. When establishing physical packages for processor subcomponents (e.g., component computer systems) the provider shall populate both Container and SystemPackaging associations. When establishing the Product instances for the subcomponent packages the provider shall populate the ProductParentChild association to the parent product. This is illustrated in Figure 47.

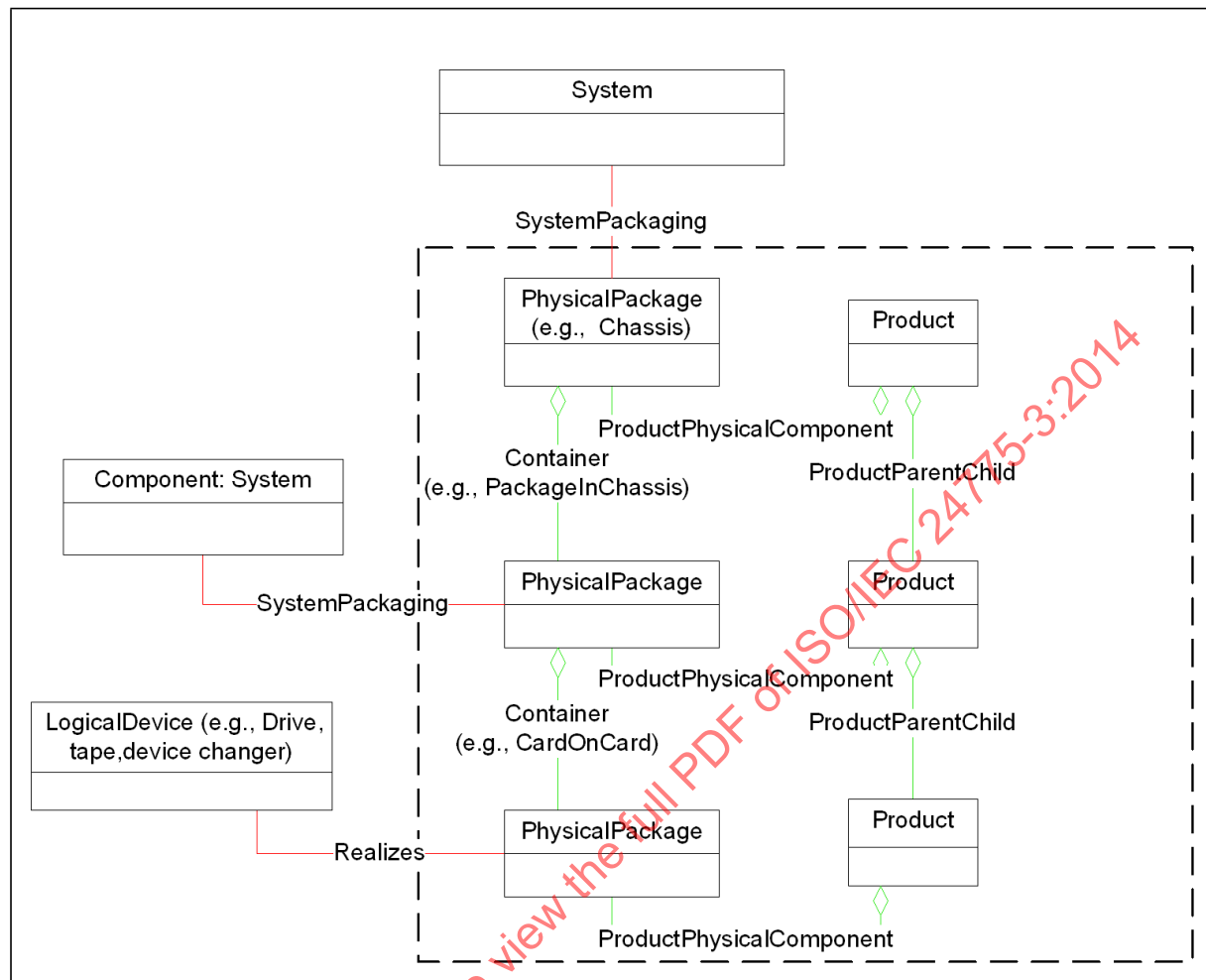


Figure 47 - Modeling for well defined subcomponents

32.1.3 Multiple Product Identities

Instrumentation may optionally describe multiple product identities for a physical package, for example, product information for both an OEM and vendor. This information should be modeled as multiple instances of `CIM_Product` associated with the `LogicalIdentity` association. The `Product` instance that clients should treat as primary is directly associated with `PhysicalPackage` via `ProductPhysicalComponent`. Additional product instances are associated with the primary product using the `LogicalIdentity` association.

Figure 48 shows an example of the use of mandatory and optional physical package classes.

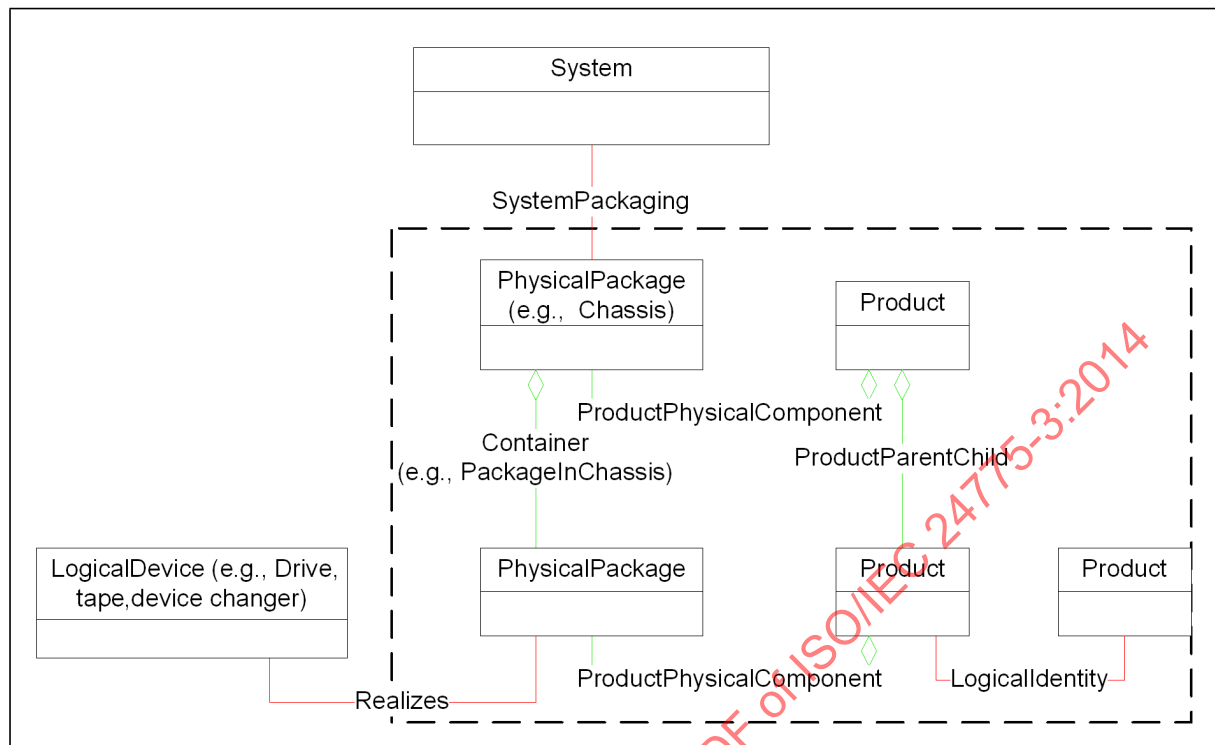


Figure 48 - Physical Package Package with Optional Classes

32.2 Health and Fault Management Considerations

Not defined in this International Standard.

32.3 Cascading Considerations

Not defined in this International Standard.

32.4 Supported Subprofiles and Packages

Related Profiles for Physical Package: Not defined in this International Standard.

32.5 Methods of this Profile

Not defined in this International Standard.

32.6 Client Considerations and Recipes

32.6.1 Find Asset Information

Information about a system is modeled in PhysicalPackage. PhysicalPackage may be subclassed to Chassis; the more general PhysicalPackage is used here to accommodate device implementations that are deployed in multiple chassis. PhysicalPackage has an associated Product with physical asset information such as Vendor and Version.

32.6.2 Finding Product information

To locate product information (Vendor, Serial number and product versions) information about a device that conforms to the profile, you would start with the “top-level” computer system and traverse the SystemPackaging to the PhysicalPackage (e.g., a Chassis). From the PhysicalPackage, the client would then traverse the ProductPhysicalComponent association to locate the Product instance. The primary Vendor, Serial Number and version for the device is in the Product instance associated with the PhysicalPackage. Additional product identities may be associated with the primary Product using the LogicalIdentity association.

32.6.3 Finding Asset information

There are certain subcomponents of a device that a client may be interested in locating. For example, disk drives in an array or changer devices in a library. To locate the asset information of these subcomponents, the client would follow the ProductParentChild association from the system Product to lower level Products.

Alternatively, if the client is starting from a LogicalDevice, it can locate the PhysicalPackage by following the Realizes association from the LogicalDevice. From the PhysicalPackage, the client can find the Product information by traversing the ProductPhysicalComponent association.

32.7 Registered Name and Version

Physical Package version 1.5.0 (Component Profile).

32.8 CIM Elements

32.8.1 General

Table 305 describes the CIM elements for Physical Package.

Table 305 - CIM Elements for Physical Package

Element Name	Requirement	Description
32.8.2 CIM_Container	Optional	Associates a PhysicalPackage to its component physical packages (e.g., Drives in a Storage System).
32.8.3 CIM_LogicalIdentity	Optional	Associates the primary product information to secondary product information.
32.8.4 CIM_PhysicalElementLocation	Conditional	Conditional requirement: Support for the Location profile. Associates the physical package of the system to its location.
32.8.5 CIM_PhysicalPackage (Component)	Optional	A physical package for a component of the overall system.
32.8.6 CIM_PhysicalPackage (System)	Mandatory	The physical package for the overall system.
32.8.7 CIM_Product (Component)	Optional	The product information for a physical package that is a component of the system.
32.8.8 CIM_Product (System)	Mandatory	The product information for the physical package of the system.
32.8.9 CIM_ProductParentChild	Optional	If more than one product comprises a system, this association should be used to indicate the 'parent' product.
32.8.10 CIM_ProductPhysicalComponent (Component)	Optional	Associates a component physical package to its product information.
32.8.11 CIM_ProductPhysicalComponent (System)	Mandatory	Associates the system physical package to its product information.
32.8.12 CIM_SystemPackaging (Component)	Optional	Associates a component system and its physical components.
32.8.13 CIM_SystemPackaging (System)	Mandatory	Associates the top level system and its physical package.

32.8.2 CIM_Container

Associates a PhysicalPackage to its component physical packages (e.g., Drives in a Storage System). This may be subclassed (e.g., PackageInChassis or CardOnCard), but only the Container properties are required.

Created By: Static
 Modified By: Static
 Deleted By: Static
 Requirement: Optional

Table 306 describes class CIM_Container.

Table 306 - SMI Referenced Properties/Methods for CIM_Container

Properties	Flags	Requirement	Description and Notes
GroupComponent		Mandatory	A reference to the higher level physical package.
PartComponent		Mandatory	A reference to a lower level physical package.

32.8.3 CIM_LogicalIdentity

Associates the primary product information to secondary product information. The secondary product information might be the OEM product information.

Created By: Static
 Modified By: Static
 Deleted By: Static
 Requirement: Optional

Table 307 describes class CIM_LogicalIdentity.

Table 307 - SMI Referenced Properties/Methods for CIM_LogicalIdentity

Properties	Flags	Requirement	Description and Notes
SystemElement		Mandatory	A reference to the primary product information.
SameElement		Mandatory	A reference to a secondary product information.

32.8.4 CIM_PhysicalElementLocation

Associates the physical package of the system to its location.

Created By: Static
 Modified By: Static
 Deleted By: Static
 Requirement: Support for the Location profile.

Table 308 describes class CIM_PhysicalElementLocation.

Table 308 - SMI Referenced Properties/Methods for CIM_PhysicalElementLocation

Properties	Flags	Requirement	Description and Notes
PhysicalLocation		Mandatory	The reference to the location of the system physical package.
Element		Mandatory	The reference to the system physical package.

32.8.5 CIM_PhysicalPackage (Component)

A physical package for a component of the overall system. There may be multiple instances of a component physical package. For certain component physical packages, this "generic" physical package may be the same as physical packages defined in other component profiles (e.g., this physical package is the same instance as the physical package defined in Disk Drive Lite). It is preferred that component physical packages are modeled in their respective component profiles. However, the component physical package defined in this profile is intended for components (e.g., non-Top Level Systems in the Multiple Computer System Profile) that do not model their physical package.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 309 describes class CIM_PhysicalPackage (Component).

Table 309 - SMI Referenced Properties/Methods for CIM_PhysicalPackage (Component)

Properties	Flags	Requirement	Description and Notes
CreationClassName		Mandatory	
Tag		Mandatory	
ElementName		Optional	
Name		Optional	
Manufacturer		Mandatory	
Model		Mandatory	
SerialNumber		Optional	
Version		Optional	
PartNumber		Optional	

32.8.6 CIM_PhysicalPackage (System)

The physical package for the overall system. There shall be only one instance of this class for an autonomous profile (e.g., Array or Fabric).

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 310 describes class CIM_PhysicalPackage (System).

Table 310 - SMI Referenced Properties/Methods for CIM_PhysicalPackage (System)

Properties	Flags	Requirement	Description and Notes
CreationClassName		Mandatory	
Tag		Mandatory	
ElementName		Optional	
Name		Optional	
Manufacturer		Mandatory	

Table 310 - SMI Referenced Properties/Methods for CIM_PhysicalPackage (System)

Properties	Flags	Requirement	Description and Notes
Model		Mandatory	
SerialNumber		Optional	
Version		Optional	
PartNumber		Optional	

32.8.7 CIM_Product (Component)

The product information for a physical package that is a component of the system.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 311 describes class CIM_Product (Component).

Table 311 - SMI Referenced Properties/Methods for CIM_Product (Component)

Properties	Flags	Requirement	Description and Notes
Name		Mandatory	
IdentifyingNumber		Mandatory	
Vendor		Mandatory	
Version		Mandatory	
ElementName		Mandatory	

32.8.8 CIM_Product (System)

The product information for the physical package of the system.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 312 describes class CIM_Product (System).

Table 312 - SMI Referenced Properties/Methods for CIM_Product (System)

Properties	Flags	Requirement	Description and Notes
Name		Mandatory	
IdentifyingNumber		Mandatory	
Vendor		Mandatory	
Version		Mandatory	
ElementName		Mandatory	

32.8.9 CIM_ProductParentChild

If more than one product comprises a system, this association should be used to indicate the 'parent' product.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 313 describes class CIM_ProductParentChild.

Table 313 - SMI Referenced Properties/Methods for CIM_ProductParentChild

Properties	Flags	Requirement	Description and Notes
Parent		Mandatory	A reference to the parent (System or Component) product.
Child		Mandatory	A reference to a component product.

32.8.10 CIM_ProductPhysicalComponent (Component)

Associates a component physical package to its product information.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 314 describes class CIM_ProductPhysicalComponent (Component).

Table 314 - SMI Referenced Properties/Methods for CIM_ProductPhysicalComponent (Component)

Properties	Flags	Requirement	Description and Notes
GroupComponent		Mandatory	A reference to the product information for a component of the system.
PartComponent		Mandatory	A reference to the component physical package.

32.8.11 CIM_ProductPhysicalComponent (System)

Associates the system physical package to its product information.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 315 describes class CIM_ProductPhysicalComponent (System).

Table 315 - SMI Referenced Properties/Methods for CIM_ProductPhysicalComponent (System)

Properties	Flags	Requirement	Description and Notes
GroupComponent		Mandatory	The reference to the product information for the system.
PartComponent		Mandatory	The reference to the system physical package.

32.8.12 CIM_SystemPackaging (Component)

Associates a component system and its physical components. The ComputerSystemPackage subclass should be used if the referenced system is subclassed as *Information Technology – Storage Management, Part 1 Overview* ComputerSystem.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 316 describes class CIM_SystemPackaging (Component).

Table 316 - SMI Referenced Properties/Methods for CIM_SystemPackaging (Component)

Properties	Flags	Requirement	Description and Notes
Dependent		Mandatory	The component system that has a physical package.
Antecedent		Mandatory	The reference to the PhysicalPackage of the component system.

32.8.13 CIM_SystemPackaging (System)

Associates the top level system and its physical package. The ComputerSystemPackage subclass should be used if the referenced system is subclassed as ComputerSystem.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 317 describes class CIM_SystemPackaging (System).

Table 317 - SMI Referenced Properties/Methods for CIM_SystemPackaging (System)

Properties	Flags	Requirement	Description and Notes
Dependent		Mandatory	The system that has a physical package.
Antecedent		Mandatory	The reference to the PhysicalPackage of the top level system.

STABLE

EXPERIMENTAL

33 Power Supply Profile

33.1 Synopsis

Profile Name: Power Supply (Component Profile)

Version: 1.0.1

Organization: SNIA

CIM Schema Version: 2.17.0

Table 318 describes the related profiles for Power Supply.

Table 318 - Related Profiles for Power Supply

Profile Name	Organization	Version	Requirement	Description
Physical Asset	DMTF	1.0.0a	Optional	

Specializes: DMTF Power Supply Profile

The SNIA Power Supply Profile specializes DSP1015: the DMTF Power Supply Profile by adding indications.

33.2 Description

The SNIA Power Supply Profile specializes the DMTF Power Supply Profile by adding indications. No other changes are made to the DMTF profile.

33.3 Implementation

33.3.1 General

See DSP1015: the DMTF Power Supply Profile.

33.3.2 Health and Fault Management Consideration

None.

33.3.3 Cascading Considerations

None.

33.4 Methods

See DSP1015: the DMTF Power Supply Profile.

33.5 Use Cases

See DSP1015: the DMTF Power Supply Profile.

33.6 CIM Elements

33.6.1 General

Table 319 describes the CIM elements for Power Supply.

Table 319 - CIM Elements for Power Supply

Element Name	Requirement	Description
33.6.2 CIM_ElementCapabilities	Conditional	Conditional requirement: Support for CIM_EnabledLogicalElementCapabilities.
CIM_EnabledLogicalElementCapabilities	Optional	
33.6.3 CIM_IsSpare	Optional	
33.6.4 CIM_MemberOfCollection	Conditional	Conditional requirement: Support for Power Supply redundancy.
33.6.5 CIM_OwningCollectionElement	Conditional	Conditional requirement: Support for Power Supply redundancy.
33.6.6 CIM_PowerSupply	Mandatory	
33.6.7 CIM_RedundancySet	Optional	
33.6.8 CIM_SuppliesPower	Optional	
33.6.9 CIM_SystemDevice	Mandatory	
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_PowerSupply	Mandatory	Creation of a PowerSupply instance.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_PowerSupply	Mandatory	Deletion of a PowerSupply instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_PowerSupply AND SourceInstance.CIM_PowerSupply::OperationalStatus <> PreviousInstance.CIM_PowerSupply::OperationalStatus	Mandatory	CQL -Change of Operational Status of a PowerSupply instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_PowerSupply AND SourceInstance.CIM_PowerSupply::EnabledState <> PreviousInstance.CIM_PowerSupply::EnabledState	Mandatory	CQL -Change of EnabledState of a PowerSupply instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_RedundancySet AND SourceInstance.CIM_RedundancySet::RedundancyStatus <> PreviousInstance.CIM_RedundancySet::RedundancyStatus	Conditional	Conditional requirement: Support for Power Supply redundancy. CQL -Change of redundancy status.

33.6.2 CIM_ElementCapabilities

CIM_ElementCapabilities is used to associate CIM_PowerSupply with CIM_EnabledLogicalElementCapabilities that describes the capabilities of CIM_PowerSupply.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for CIM_EnabledLogicalElementCapabilities.

Table 320 describes class CIM_ElementCapabilities.

Table 320 - SMI Referenced Properties/Methods for CIM_ElementCapabilities

Properties	Requirement	Description and Notes
ManagedElement	Mandatory	Reference to CIM_PowerSupply.
Capabilities	Mandatory	Reference to CIM_EnabledLogicalElementCapabilities.

CIM_EnabledLogicalElementCapabilities

CIM_EnabledLogicalElementCapabilities represents the capabilities of the power supply.

Requirement: Optional

Table 321 describes class CIM_EnabledLogicalElementCapabilities.

Table 321 - SMI Referenced Properties/Methods for CIM_EnabledLogicalElementCapabilities

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
RequestedStatesSupported	Mandatory	Array that contains the supported requested states for the instance of CIM_PowerSupply. Shall include 2 (Enabled), 3 (Disabled), 6 (Offline), or 11 (Reset).
ElementNameEditSupported	Mandatory	
MaxElementNameLen	Conditional	Conditional requirement: Support for Element Name editing. Conditional on Support for Element Name editing.
ElementName	Mandatory	User-friendly name.

33.6.3 CIM_IsSpare

CIM_IsSpare is used to associate CIM_PowerSupply with CIM_RedundancySet that the CIM_PowerSupply is a member of and where CIM_PowerSupply represents a spare power supply.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 322 describes class CIM_IsSpare.

Table 322 - SMI Referenced Properties/Methods for CIM_IsSpare

Properties	Requirement	Description and Notes
SpareStatus	Mandatory	Shall be 0 (Unknown), 1 (Cold Standby), or 2 (Hot Standby).
FailoverSupported	Mandatory	Shall be 2 (Automatic), 3 (Manual) or 4 (Both Manual and Automatic).
Antecedent	Mandatory	The RedundancySet.
Dependent	Mandatory	PowerSupply.

33.6.4 CIM_MemberOfCollection

CIM_MemberOfCollection is used to associate CIM_PowerSupply with CIM_RedundancySet that the CIM_PowerSupply is a member of.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for Power Supply redundancy.

Table 323 describes class CIM_MemberOfCollection.

Table 323 - SMI Referenced Properties/Methods for CIM_MemberOfCollection

Properties	Requirement	Description and Notes
Collection	Mandatory	
Member	Mandatory	

33.6.5 CIM_OwningCollectionElement

CIM_OwningCollectionElement is used to associate CIM_RedundancySet with CIM_ComputerSystem that the CIM_RedundancySet is a member of. The instance of CIM_OwningCollectionElement is conditional on having instantiation of the CIM_RedundancySet class.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for Power Supply redundancy.

Table 324 describes class CIM_OwningCollectionElement.

Table 324 - SMI Referenced Properties/Methods for CIM_OwningCollectionElement

Properties	Requirement	Description and Notes
OwnedElement	Mandatory	
OwningElement	Mandatory	

33.6.6 CIM_PowerSupply

CIM_PowerSupply is used to represent the power supply.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 325 describes class CIM_PowerSupply.

Table 325 - SMI Referenced Properties/Methods for CIM_PowerSupply

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	Key.
SystemName	Mandatory	Key.
CreationClassName	Mandatory	Key.
DeviceID	Mandatory	Key.
TotalOutputPower	Mandatory	Shall match 0 when the power supply's total output power is unknown.
ElementName	Mandatory	
OperationalStatus	Mandatory	
HealthState	Mandatory	
EnabledState	Mandatory	Shall be 2 (Enabled), 3 (Disabled), 5 (Not Applicable) or 6 (Enabled but Offline).
RequestedState	Mandatory	Shall be 2 (Enabled), 3 (Disabled), 5 (No Change), 6 (Offline), 11 (Reset) or 12 (Not Applicable).
RequestStateChange()	Conditional	Conditional requirement: updating requested states. The implementation shall support this method, but the method may always return 'Not Supported'.

33.6.7 CIM_RedundancySet

CIM_RedundancySet is used to represent the aggregation of redundant power supplies.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 326 describes class CIM_RedundancySet.

Table 326 - SMI Referenced Properties/Methods for CIM_RedundancySet

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
ElementName	Mandatory	Shall be formatted as a free formed string of variable length (pattern ".+").
RedundancyStatus	Mandatory	
TypeOfSet	Mandatory	Shall be 2 (N + 1), 3 (Load Balanced), 4 (Sparing) or 5 (Limited Sparing).
MinNumberNeeded	Mandatory	shall match 0 when the minimum number of power supplies needed for the redundancy is unknown.
Failover()	Optional	

33.6.8 CIM_SuppliesPower

CIM_SuppliesPower is used to associate CIM_PowerSupply with CIM_ManagedSystemElement that the power supply represented by the CIM_PowerSupply instance supplies power to.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 327 describes class CIM_SuppliesPower.

Table 327 - SMI Referenced Properties/Methods for CIM_SuppliesPower

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	Shall reference the instance of the subclass of CIM_ManagedSystemElement representing element receiving the power.

33.6.9 CIM_SystemDevice

CIM_SystemDevice is used to associate CIM_PowerSupply with CIM_ComputerSystem that the CIM_PowerSupply is a member of.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 328 describes class CIM_SystemDevice.

Table 328 - SMI Referenced Properties/Methods for CIM_SystemDevice

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	
PartComponent	Mandatory	

EXPERIMENTAL

EXPERIMENTAL

34 Fan Profile

34.1 Synopsis

Profile Name: Fan (Component Profile)

Version: 1.0.1

Organization: SNIA

CIM Schema Version: 2.11.0

Table 329 describes the related profiles for Fan.

Table 329 - Related Profiles for Fan

Profile Name	Organization	Version	Requirement	Description
Physical Asset	DMTF	1.0.0a	Optional	
Sensors	SNIA	1.0.0	Optional	

Specializes: DMTF Fan Profile

The SNIA Fan Profile specializes DSP1013: the DMTF Fan Profile by adding indications.

34.2 Description

The SNIA Fan Profile specializes the DMTF Fan Profile by adding indications. No other changes are made to the DMTF Profile.

34.3 Implementation

See DSP1013: the DMTF Fan Profile.

34.3.1 Health and Fault Management Consideration

None.

34.3.2 Cascading Considerations

None.

34.4 Methods

See DSP1013: the DMTF Fan Profile.

34.5 Use Cases

See DSP1013: the DMTF Fan Profile.

34.6 CIM Elements

34.6.1 General

Table 330 describes the CIM elements for Fan.

Table 330 - CIM Elements for Fan

Element Name	Requirement	Description
34.6.2 CIM_AssociatedCooling	Optional	
34.6.3 CIM_AssociatedSensor	Optional	
34.6.4 CIM_ElementCapabilities	Conditional	Conditional requirement: Support for CIM_EnabledLogicalElementCapabilities.
34.6.5 CIM_EnabledLogicalElementCapabilities	Optional	
34.6.6 CIM_Fan	Mandatory	
34.6.7 CIM_IsSpare	Optional	
34.6.8 CIM_MemberOfCollection	Conditional	Conditional requirement: Support for Fan redundancy.
34.6.9 CIM_NumericSensor	Optional	
34.6.10 CIM_OwningCollectionElement	Conditional	Conditional requirement: Support for Fan redundancy.
34.6.11 CIM_RedundancySet	Optional	
34.6.12 CIM_Sensor	Optional	
34.6.13 CIM_SystemDevice	Mandatory	
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_Fan	Mandatory	Creation of a Fan instance.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_Fan	Mandatory	CQL -Deletion of a Fan instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_Fan AND SourceInstance.CIM_Fan::OperationalStatus <> PreviousInstance.CIM_Fan::OperationalStatus	Mandatory	CQL -Change of Operational Status of a Fan instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_Fan AND SourceInstance.CIM_Fan::EnabledState <> PreviousInstance.CIM_Fan::EnabledState	Mandatory	CQL -Change of EnabledState of a Fan instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_RedundancySet AND SourceInstance.CIM_RedundancySet::RedundancyStatus <> PreviousInstance.CIM_RedundancySet::RedundancyStatus	Conditional	Conditional requirement: Support for Fan redundancy. CQL -Change of redundancy status.

34.6.2 CIM_AssociatedCooling

CIM_AssociatedCooling associates CIM_Fan with a subclass of CIM_ManagedSystemElement.

Requirement: Optional.

Table 331 describes class CIM_AssociatedCooling.

Table 331 - SMI Referenced Properties/Methods for CIM_AssociatedCooling

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to CIM_Fan.
Dependent	Mandatory	Shall reference an instance of a subclass of CIM_ManagedSystemElement for which the fan is providing cooling.

34.6.3 CIM_AssociatedSensor

Requirement: Optional.

Table 332 describes class CIM_AssociatedSensor.

Table 332 - SMI Referenced Properties/Methods for CIM_AssociatedSensor

Properties	Requirement	Description and Notes
Dependent	Mandatory	Reference to CIM_Fan.
Antecedent	Mandatory	Reference to Sensor or NumericSensor.

34.6.4 CIM_ElementCapabilities

CIM_ElementCapabilities is used to associate CIM_Fan with the CIM_EnabledLogicalElementCapabilities instance that describes the capabilities of the fan.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for CIM_EnabledLogicalElementCapabilities.

Table 333 describes class CIM_ElementCapabilities.

Table 333 - SMI Referenced Properties/Methods for CIM_ElementCapabilities

Properties	Requirement	Description and Notes
ManagedElement	Mandatory	Reference to CIM_Fan.
Capabilities	Mandatory	Reference to CIM_EnabledLogicalElementCapabilities.

34.6.5 CIM_EnabledLogicalElementCapabilities

CIM_EnabledLogicalElementCapabilities represents the capabilities of the Fan.

Requirement: Optional.

Table 334 describes class CIM_EnabledLogicalElementCapabilities.

Table 334 - SMI Referenced Properties/Methods for CIM_EnabledLogicalElementCapabilities

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
RequestedStatesSupported	Mandatory	Array that contains the supported requested states for the instance of CIM_Fan. Shall be an empty array or contain any combination of: 2 (Enabled), 3 (Disabled), or 11 (Reset).
ElementNameEditSupported	Mandatory	
MaxElementNameLen	Conditional	Conditional requirement: Support for Element Name editing. Conditional on Support for Element Name editing.

34.6.6 CIM_Fan

CIM_Fan is used to represent the fan.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 335 describes class CIM_Fan.

Table 335 - SMI Referenced Properties/Methods for CIM_Fan

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	Key.
SystemName	Mandatory	Key.
CreationClassName	Mandatory	Key.
DeviceID	Mandatory	Key.
ElementName	Mandatory	
OperationalStatus	Mandatory	
HealthState	Mandatory	
EnabledState	Mandatory	
VariableSpeed	Mandatory	
DesiredSpeed	Conditional	Conditional requirement: Support for the SetSpeed method.
ActiveCooling	Mandatory	Shall have the value TRUE.
RequestedState	Mandatory	
SetSpeed()	Conditional	Conditional requirement: Support for the SetSpeed method.
RequestStateChange()	Mandatory	The implementation shall support this method, but the method may always return 'Not Supported.'

34.6.7 CIM_IsSpare

CIM_IsSpare is used to associate CIM_Fan with CIM_RedundancySet that the CIM_Fan is a member of and where CIM_Fan represents a spare Fan.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 336 describes class CIM_IsSpare.

Table 336 - SMI Referenced Properties/Methods for CIM_IsSpare

Properties	Requirement	Description and Notes
SpareStatus	Mandatory	Shall be 0 (Unknown) or 3 (Cold Standby).
FailoverSupported	Mandatory	Shall be 2 (Automatic), 3 (Manual) or 4 (Both Manual and Automatic).
Antecedent	Mandatory	The RedundancySet.
Dependent	Mandatory	The Fan.

34.6.8 CIM_MemberOfCollection

CIM_MemberOfCollection is used to associate CIM_Fan with CIM_RedundancySet that the CIM_Fan is a member of.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for Fan redundancy.

Table 337 describes class CIM_MemberOfCollection.

Table 337 - SMI Referenced Properties/Methods for CIM_MemberOfCollection

Properties	Requirement	Description and Notes
Collection	Mandatory	Reference to RedundancySet.
Member	Mandatory	Reference to CIM_Fan.

34.6.9 CIM_NumericSensor

The CIM_NumericSensor class is defined by the Sensors Profile.

Requirement: Optional.

Table 338 describes class CIM_NumericSensor.

Table 338 - SMI Referenced Properties/Methods for CIM_NumericSensor

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	Key.
SystemName	Mandatory	Key.
CreationClassName	Mandatory	Key.

Table 338 - SMI Referenced Properties/Methods for CIM_NumericSensor

Properties	Requirement	Description and Notes
DeviceID	Mandatory	Key.
SensorType	Mandatory	Shall be set to 5 (Tachometer).
BaseUnits	Mandatory	Shall be 19 (RPM).
RateUnits	Mandatory	Shall be 0 (None).

34.6.10 CIM_OwningCollectionElement

CIM_OwningCollectionElement is used to associate CIM_RedundancySet with CIM_ComputerSystem that the CIM_RedundancySet is a member of. The instance of CIM_OwningCollectionElement is conditional on having instantiation of the CIM_RedundancySet class.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for Fan redundancy.

Table 339 describes class CIM_OwningCollectionElement.

Table 339 - SMI Referenced Properties/Methods for CIM_OwningCollectionElement

Properties	Requirement	Description and Notes
OwnedElement	Mandatory	Reference to RedundancySet.
OwningElement	Mandatory	Reference to CComputerSystem in referencing profile.

34.6.11 CIM_RedundancySet

CIM_RedundancySet is used to represent the aggregation of redundant fan.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 340 describes class CIM_RedundancySet.

Table 340 - SMI Referenced Properties/Methods for CIM_RedundancySet

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
ElementName	Mandatory	Shall be formatted as a free formed string of variable length (pattern ".+").
RedundancyStatus	Mandatory	Shall be 2 (Fully Redundant), 3 (Degraded Redundancy), 4 (Redundancy Lost) or 5 (Overall Failure).

Table 340 - SMI Referenced Properties/Methods for CIM_RedundancySet

Properties	Requirement	Description and Notes
TypeOfSet	Mandatory	Shall be 2 (N+1), 3 (Load Balanced), 4 (Sparing) or 5 (Limited Sparing).
MinNumberNeeded	Mandatory	Shall be 0 when the minimum number of fans needed for the redundancy is unknown.
Failover()	Optional	

34.6.12 CIM_Sensor

The CIM_Sensor class is defined by the Sensors Profile.

Requirement: Optional.

Table 341 describes class CIM_Sensor.

Table 341 - SMI Referenced Properties/Methods for CIM_Sensor

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	Key.
SystemName	Mandatory	Key.
CreationClassName	Mandatory	Key.
DeviceID	Mandatory	Key.
SensorType	Mandatory	Shall be set to 5 (Tachometer).

34.6.13 CIM_SystemDevice

CIM_SystemDevice is used to associate CIM_Fan with CIM_ComputerSystem that the CIM_Fan is a member of.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 342 describes class CIM_SystemDevice.

Table 342 - SMI Referenced Properties/Methods for CIM_SystemDevice

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	
PartComponent	Mandatory	

EXPERIMENTAL

EXPERIMENTAL

35 Sensors Profile

35.1 Synopsis

Profile Name: Sensors (Component Profile)

Version: 1.0.0

Organization: SNIA

CIM Schema Version: 2.11.0

Related Profiles for Sensors: Not defined in this International Standard.

Specializes: DMTF Sensors Profile

The SNIA Sensors Profile specializes DSP1009: the DMTF Sensors Profile by adding indications.

35.2 Description

The SNIA Sensors Profile specializes the DMTF Sensors Profile by adding indications. No other changes are made to the DMTF Profile.

35.3 Implementation

35.3.1 General

See DSP1009: the DMTF Sensors Profile.

35.3.2 Health and Fault Management Consideration

None.

35.3.3 Cascading Considerations

None.

35.4 Methods

See DSP1009: the DMTF Sensors Profile.

35.5 Use Cases

See DSP1009: the DMTF Sensors Profile.

35.6 CIM Elements

35.6.1 General

Table 343 describes the CIM elements for Sensors.

Table 343 - CIM Elements for Sensors

Element Name	Requirement	Description
35.6.2 CIM_AssociatedSensor	Optional	
35.6.3 CIM_ElementCapabilities	Conditional	Conditional requirement: EnabledLogicalElementCapabilities
35.6.4 CIM_EnabledLogicalElementCapabilities	Optional	
35.6.5 CIM_NumericSensor	Conditional	Conditional requirement: Absence of Support for CIM_Sensor.

Table 343 - CIM Elements for Sensors

Element Name	Requirement	Description
35.6.6 CIM_Sensor	Conditional	Conditional requirement: Absence of Support for CIM_NumericSensor.
35.6.7 CIM_SystemDevice	Mandatory	
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_Sensor	Mandatory	Creation of a Sensor instance.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_Sensor	Mandatory	Deletion of a Sensor instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_Sensor AND SourceInstance.CIM_Sensor::OperationalStatus <> PreviousInstance.CIM_Sensor::OperationalStatus	Mandatory	CQL -Change of Operational Status of a Sensor instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_Sensor AND SourceInstance.CIM_Sensor::EnabledState <> PreviousInstance.CIM_Sensor::EnabledState	Mandatory	CQL -Change of EnabledState of a Sensor instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_Sensor AND SourceInstance.CIM_Sensor::CurrentState <> PreviousInstance.CIM_Sensor::CurrentState	Mandatory	CQL -Change of Current State of a Sensor instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_NumericSensor AND SourceInstance.CIM_Sensor::CurrentReading <> PreviousInstance.CIM_Sensor::CurrentReading	Mandatory	CQL -Change of Current Reading of a Sensor instance.

35.6.2 CIM_AssociatedSensor

CIM_AssociatedSensor associates CIM_Sensor or CIM_NumericSensor with a subclass of CIM_ManagedSystemElement.

Requirement: Optional.

Table 344 describes class CIM_AssociatedSensor.

Table 344 - SMI Referenced Properties/Methods for CIM_AssociatedSensor

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Shall be a reference to a specific instance of CIM_Sensor or CIM_NumericSensor.
Dependent	Mandatory	Shall reference an instance of a subclass of CIM_ManagedSystemElement for which the sensor is monitoring.

35.6.3 CIM_ElementCapabilities

CIM_ElementCapabilities is used to associate CIM_Sensor with the CIM_EnabledLogicalElementCapabilities instance that describes the capabilities of the fan.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: null.

Table 345 describes class CIM_ElementCapabilities.

Table 345 - SMI Referenced Properties/Methods for CIM_ElementCapabilities

Properties	Requirement	Description and Notes
ManagedElement	Mandatory	
Capabilities	Mandatory	

35.6.4 CIM_EnabledLogicalElementCapabilities

CIM_EnabledLogicalElementCapabilities represents the capabilities of the Fan.

Requirement: Optional.

Table 346 describes class CIM_EnabledLogicalElementCapabilities.

Table 346 - SMI Referenced Properties/Methods for CIM_EnabledLogicalElementCapabilities

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
RequestedStatesSupported	Mandatory	Array that contains the supported requested states for the instance of CIM_Sensor.
ElementNameEditSupported	Mandatory	
MaxElementNameLen	Conditional	Conditional requirement: EditSupportConditional on Support for Element Name editing.

35.6.5 CIM_NumericSensor

CIM_NumericSensor is used to represent an analog sensor. The CIM_NumericSensor class is mandatory when the CIM_Sensor class is not implemented.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Absence of Support for CIM_Sensor.

Table 347 describes class CIM_NumericSensor.

Table 347 - SMI Referenced Properties/Methods for CIM_NumericSensor

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	Key.
SystemName	Mandatory	Key.
CreationClassName	Mandatory	Key.
DeviceID	Mandatory	Key.
BaseUnits	Mandatory	
UnitModifier	Mandatory	
RateUnits	Mandatory	
CurrentReading	Mandatory	
LowerThresholdNonCritical	Conditional	See DMTF Sensors Profile.

Table 347 - SMI Referenced Properties/Methods for CIM_NumericSensor

Properties	Requirement	Description and Notes
UpperThresholdNonCritical	Conditional	See DMTF Sensors Profile.
LowerThresholdCritical	Conditional	See DMTF Sensors Profile.
UpperThresholdCritical	Conditional	See DMTF Sensors Profile.
LowerThresholdFatal	Conditional	See DMTF Sensors Profile.
UpperThresholdFatal	Conditional	See DMTF Sensors Profile.
SupportedThresholds	Mandatory	See DMTF Sensors Profile.
SettableThresholds	Mandatory	See DMTF Sensors Profile.
SensorType	Mandatory	
PossibleStates	Mandatory	
CurrentState	Mandatory	
ElementName	Mandatory	
OtherSensorTypeDescription	Conditional	Conditional requirement: The OtherSensorTypeDescription property shall be mandatory when the SensorType property is set to a value of 1 (Other). The OtherSensorTypeDescription property shall be formatted as a free-formed string of variable length (pattern \.*\').
EnabledState	Mandatory	
RequestedState	Mandatory	
OperationalStatus	Mandatory	
HealthState	Mandatory	
RequestStateChange()	Conditional	Conditional requirement: When a CIM_EnabledLogicalElementCapabilities instance is associated with the Central Instance and the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property is a non-empty array, sensor state management shall be supported.
RestoreDefaultThresholds()	Conditional	Conditional requirement: The CIM_NumericSensor.RestoreDefaultThresholds() method shall be implemented and shall not return a value of 1 (Unsupported) when the CIM_NumericSensor.SettableThresholds property is a non-empty array.

35.6.6 CIM_Sensor

CIM_Sensor is used to represent a discrete sensor. The CIM_Sensor class is mandatory if the CIM_NumericSensor class is not implemented.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Absence of Support for CIM_NumericSensor.

Table 348 describes class CIM_Sensor.

Table 348 - SMI Referenced Properties/Methods for CIM_Sensor

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	Key.
SystemName	Mandatory	Key.
CreationClassName	Mandatory	Key.
DeviceID	Mandatory	Key.
SensorType	Mandatory	
PossibleStates	Mandatory	See DMTF Sensors Profile.
CurrentState	Mandatory	See DMTF Sensors Profile.
ElementName	Mandatory	See DMTF Sensors Profile.
OtherSensorTypeDescription	Conditional	Conditional requirement: The OtherSensorTypeDescription property shall be mandatory when the SensorType property is set to a value of 1 (Other). The OtherSensorTypeDescription property shall be formatted as a free-formed string of variable length (pattern \.*\'). See DMTF Sensors Profile.
EnabledState	Mandatory	See DMTF Sensors Profile.
RequestedState	Mandatory	See DMTF Sensors Profile.
OperationalStatus	Mandatory	
HealthState	Mandatory	
RequestStateChange()	Conditional	Conditional requirement: When a CIM_EnabledLogicalElementCapabilities instance is associated with the Central Instance and the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property is a non-empty array, sensor state management shall be supported.

35.6.7 CIM_SystemDevice

CIM_SystemDevice is used to associate CIM_Sensor with CIM_ComputerSystem that the CIM_Sensor is a member of.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 349 describes class CIM_SystemDevice.

Table 349 - SMI Referenced Properties/Methods for CIM_SystemDevice

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	
PartComponent	Mandatory	

EXPERIMENTAL

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

EXPERIMENTAL

36 Base Server Profile

36.1 Synopsis

Profile Name: Base Server (Autonomous Profile)

Version: 1.5.0

Organization: SNIA

CIM Schema Version: 2.20.0

Table 350 describes the related profiles for Base Server.

Table 350 - Related Profiles for Base Server

Profile Name	Organization	Version	Requirement	Description
Fan	DMTF	1.0.1	Optional	See DSP1013 version 1.0.1
Physical Asset	DMTF	1.0.0a	Mandatory	See DSP1004 version 1.0.1, section 5
Power Supply	DMTF	1.0.1	Optional	See DSP1015 version 1.0.1
Record Log	DMTF	1.0.1	Optional	See DSP1052 version 1.0.0, section 5
Sensors	DMTF	1.0.2	Optional	See DSP1009 version 1.0.2
Software Inventory	DMTF	1.0.1	Optional	See DSP1023 version 1.0.1
Software Update	DMTF	1.0.0	Optional	See DSP1052 version 1.0.0, section 5
Storage HBA	SNIA	1.4.0	Optional	Experimental.
Host Hardware RAID Controller	SNIA	1.3.0	Optional	Experimental.
Storage Enclosure	SNIA	1.3.0	Optional	Experimental.
Launch In Context	DMTF	1.0.0	Optional	Experimental. See DSP1102 version 1.0.0
Host Filesystem	SNIA	1.5.0	Optional	Experimental.

Specializes: DMTF Base Server 1.0.0

Central Class: CIM_ComputerSystem

Scoping Class: CIM_ComputerSystem

The Base Server Profile models a customer server or storage system.

36.2 Description

The SNIA Base Server Profile models a customer server or storage system containing storage elements. This profile may be used to scope one or more HBAs (or other storage elements).

This profile may represent either a physical system or a virtual system.

36.3 Implementation

36.3.1 General

See DSP1004, DMTF Base Server Profile for details on the model.

In a storage context, there are several related deployment options.

36.3.2 HBA Instrumentation

If an HBA vendor wishes to create HBA instrumentation that can be used with CIM instrumentation from a server vendor, they would implement the component Storage HBA Profile and work with the server vendor(s) to assure it integrates effectively with their autonomous server profile. If an HBA vendor wishes to deliver a free-standing implementation that does not rely on server-vendor software, they could implement this profile along with the Storage HBA Profile. Note that the HBA vendor could support both approaches and let a customer or installation script determine which is most appropriate.

36.3.3 Host Hardware RAID Instrumentation

Host Hardware RAID vendors have the same deployment options as HBA vendors (see 36.3.2)

36.3.4 Storage Enclosure Instrumentation

In configurations where the Storage Enclosure Profile is not used with a single autonomous profile, the Base Server may be used as the referencing profile for the Storage Enclosure and other component profiles.

36.3.5 Health and Fault Management Consideration

Not defined in this International Standard.

36.3.6 Cascading Considerations

None.

36.4 Methods

See DSP1004, DMTF Base Server Profile.

36.5 Use Cases

See DSP1004, DMTF Base Server Profile.

36.6 CIM Elements

36.6.1 General

Table 351 describes the CIM elements for Base Server.

Table 351 - CIM Elements for Base Server

Element Name	Requirement	Description
36.6.2 CIM_ComputerSystem	Mandatory	The hosting system for the Storage Elements. Associated to RegisteredProfile.
36.6.3 CIM_ComputerSystemPackage	Mandatory	DSP1004 version 1.0.1, section 10.2.
36.6.4 CIM_ElementCapabilities	Optional	See DSP1052 version 1.0.0, section 10.2.
36.6.5 CIM_EnabledLogicalElementCapabilities	Optional	See DSP1052 version 1.0.0, section 10.3 and DSP1004 version 1.0.1, section 10.2.
36.6.6 CIM_HostedService	Optional	See DSP1052 version 1.0.0, section 10.4.
36.6.7 CIM_PhysicalPackage	Mandatory	DSP1004 version 1.0.1, section 10.4 and DSP1011 version 1.0.1, section 10.16.
36.6.8 CIM_ServiceAffectsElement	Optional	See DSP1052 version 1.0.0, section 10.5.
36.6.9 CIM_TimeService	Optional	Experimental. See DSP1052 version 1.0.0, section 10.6.

36.6.2 CIM_ComputerSystem

The hosting system for the Storage Elements. The class definition specializes the CIM_ComputerSystem definition in the Base Server profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Shall be associated to RegisteredProfile using ElementConformsToProfile association. The RegisteredProfile instance shall have RegisteredName set to 'Base Server', RegisteredOrganization set to 'SNIA', and RegisteredVersion set to '1.5.0'.

Table 352 describes class CIM_ComputerSystem.

Table 352 - SMI Referenced Properties/Methods for CIM_ComputerSystem

Properties	Requirement	Description and Notes
CreationClassName	Mandatory	Key.
Name (overridden)	Mandatory	Unique identifier for the hosting system.
ElementName (overridden)	Mandatory	User friendly name.
OtherIdentifyingInfo (overridden)	Mandatory	
IdentifyingDescriptions	Optional	See DSP1052 version 1.0.0, section 10.1.
Dedicated (overridden)	Mandatory	0 (Not Dedicated).
OperationalStatus	Mandatory	See DSP1052 version 1.0.0, section 10.1.
HealthState	Mandatory	See DSP1052 version 1.0.0, section 10.1.
EnabledState	Mandatory	See DSP1052 version 1.0.0, section 10.1 and DSP1004 version 1.0.1, section 10.1.
RequestedState	Mandatory	See DSP1052 version 1.0.0, section 10.1 and DSP1004 version 1.0.1, section 10.1.
NameFormat (added)	Mandatory	
OtherDedicatedDescriptions (added)	Optional	
RequestStateChange()	Conditional	Conditional requirement: The CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property contains at least one value. See DSP1052 version 1.0.0, section 10.1.

36.6.3 CIM_ComputerSystemPackage

One or more instances of CIM_ComputerSystemPackage associate the CIM_ComputerSystem instance with the CIM_PhysicalPackage instances in which it resides. The constraints specified in this Table are in addition to those specified in the Physical Asset Profile.

Requirement: Mandatory.

Table 353 describes class CIM_ComputerSystemPackage.

Table 353 - SMI Referenced Properties/Methods for CIM_ComputerSystemPackage

Properties	Requirement	Description and Notes
Dependent	Mandatory	
Antecedent	Mandatory	

36.6.4 CIM_ElementCapabilities

CIM_ElementCapabilities associates an instance of CIM_EnabledLogicalElementCapabilities with an instance of CIM_ComputerSystem.

Requirement: Optional.

Table 354 describes class CIM_ElementCapabilities.

Table 354 - SMI Referenced Properties/Methods for CIM_ElementCapabilities

Properties	Requirement	Description and Notes
ManagedElement	Mandatory	This property shall be a reference to an instance of CIM_ComputerSystem.
Capabilities	Mandatory	This property shall be a reference to the instance of CIM_EnabledLogicalElementCapabilities.

36.6.5 CIM_EnabledLogicalElementCapabilities

CIM_EnabledLogicalElementCapabilities indicates support for managing the state of the system.

Requirement: Optional.

Table 355 describes class CIM_EnabledLogicalElementCapabilities.

Table 355 - SMI Referenced Properties/Methods for CIM_EnabledLogicalElementCapabilities

Properties	Requirement	Description and Notes
InstanceID	Mandatory	Key.
RequestedStatesSupported	Mandatory	See DSP1052 version 1.0.0, section 10.3 and DSP1004 version 1.0.1, section 10.3.
ElementNameEditSupported	Mandatory	See DSP1052 version 1.0.0, section 10.3.
MaxElementNameLen	Optional	See DSP1052 version 1.0.0, section 10.3.

36.6.6 CIM_HostedService

CIM_HostedService relates the CIM_TimeService to its scoping CIM_ComputerSystem instance.

Requirement: Optional.

Table 356 describes class CIM_HostedService.

Table 356 - SMI Referenced Properties/Methods for CIM_HostedService

Properties	Requirement	Description and Notes
Antecedent	Mandatory	This property shall reference the Central Instance.
Dependent	Mandatory	This property shall reference CIM_TimeService.

36.6.7 CIM_PhysicalPackage

One or more instances of CIM_PhysicalPackage represent the physical packaging of the computer system. Other than the existence of at least one instance of CIM_PhysicalPackage, this profile does not specify any constraints for CIM_PhysicalPackage beyond those specified in the Physical Asset Profile.

Requirement: Mandatory.

Table 357 describes class CIM_PhysicalPackage.

Table 357 - SMI Referenced Properties/Methods for CIM_PhysicalPackage

Properties	Requirement	Description and Notes
Tag	Mandatory	Key.
CreationClassName	Mandatory	Key.
PackageType	Mandatory	See DSP1011 version 1.0.1, section 10.16.
Manufacturer	Conditional	Conditional requirement: The CIM_PhysicalAssetCapabilities.FRUInfoSupported has a value of TRUE. See DSP1011 version 1.0.1, section 10.16.
Model	Conditional	Conditional requirement: The CIM_PhysicalAssetCapabilities.FRUInfoSupported has a value of TRUE. See DSP1011 version 1.0.1, section 10.16.
SerialNumber	Conditional	Conditional requirement: The CIM_PhysicalAssetCapabilities.FRUInfoSupported has a value of TRUE. See DSP1011 version 1.0.1, section 10.16.
PartNumber	Conditional	Conditional requirement: The CIM_PhysicalAssetCapabilities.FRUInfoSupported has a value of TRUE. See DSP1011 version 1.0.1, section 10.16.
SKU	Conditional	Conditional requirement: The CIM_PhysicalAssetCapabilities.FRUInfoSupported has a value of TRUE. See DSP1011 version 1.0.1, section 10.16.
VendorCompatibilityStrings	Optional	See DSP1011 version 1.0.1, section 10.16.
CanBeFRUed	Optional	This property should be implemented when the Physical Element can be replaced in the field.
Version	Optional	The property shall be the hardware version.
Name	Optional	
ElementName	Mandatory	This property shall be formatted as a free-form string of variable length.

36.6.8 CIM_ServiceAffectsElement

CIM_ServiceAffectsElement associates the CIM_TimeService instance with the Central Instance.

Requirement: Optional.

Table 358 describes class CIM_ServiceAffectsElement.

Table 358 - SMI Referenced Properties/Methods for CIM_ServiceAffectsElement

Properties	Requirement	Description and Notes
ElementEffects	Mandatory	Matches 5 (Manages).
AffectedElement	Mandatory	This property shall be a reference to the Central Instance.
AffectingElement	Mandatory	This property shall be a reference to an instance of CIM_TimeService.

36.6.9 CIM_TimeService

Experimental. CIM_TimeService manages the current time on the system.

Requirement: Optional

Table 359 describes class CIM_TimeService.

Table 359 - SMI Referenced Properties/Methods for CIM_TimeService

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	Key.
SystemName	Mandatory	Key.
CreationClassName	Mandatory	Key.
Name	Mandatory	Key.
ElementName	Mandatory	See DSP1052 version 1.0.0, section 10.6.
ManageTime()	Mandatory	See DSP1052 version 1.0.0, section 10.6.

EXPERIMENTAL

EXPERIMENTAL

37 Media Access Device Profile

37.1 Synopsis

Profile Name: Media Access Device (Component Profile)

Version: 1.0.0

Organization: SNIA

CIM Schema Version: 2.11.0

Table 360 describes the related profiles for Media Access Device.

Table 360 - Related Profiles for Media Access Device

Profile Name	Organization	Version	Requirement	Description
Software Inventory	SNIA	1.0.0	Mandatory	
Software Update	DMTF	1.0.0	Optional	
Indication	SNIA	1.5.0	Optional	

The Media Access Device Profile models media access devices - such as tape and CD drives.

37.2 Description

37.2.1 General

The Media Access Device Profile models media access devices - such as tape and CD drives.

37.2.2 Location Indicator

The implementation may optionally support a drive location indicator (such as an LED) using CIM_MediaAccessDevice.LocationIndicator. The client may set this to 2 (On) or 3 (Off)). If the implementation does not support this feature, LocationIndicator shall have the value 4 (Not Supported).

37.2.3 Media Access Device Online/Offline

The drive may be started or stopped by setting the Starting and Stopping values in OperationalStatus using the RequestStateChange method. Figure 49 shows Media Access Device Class information.

See Table 361.

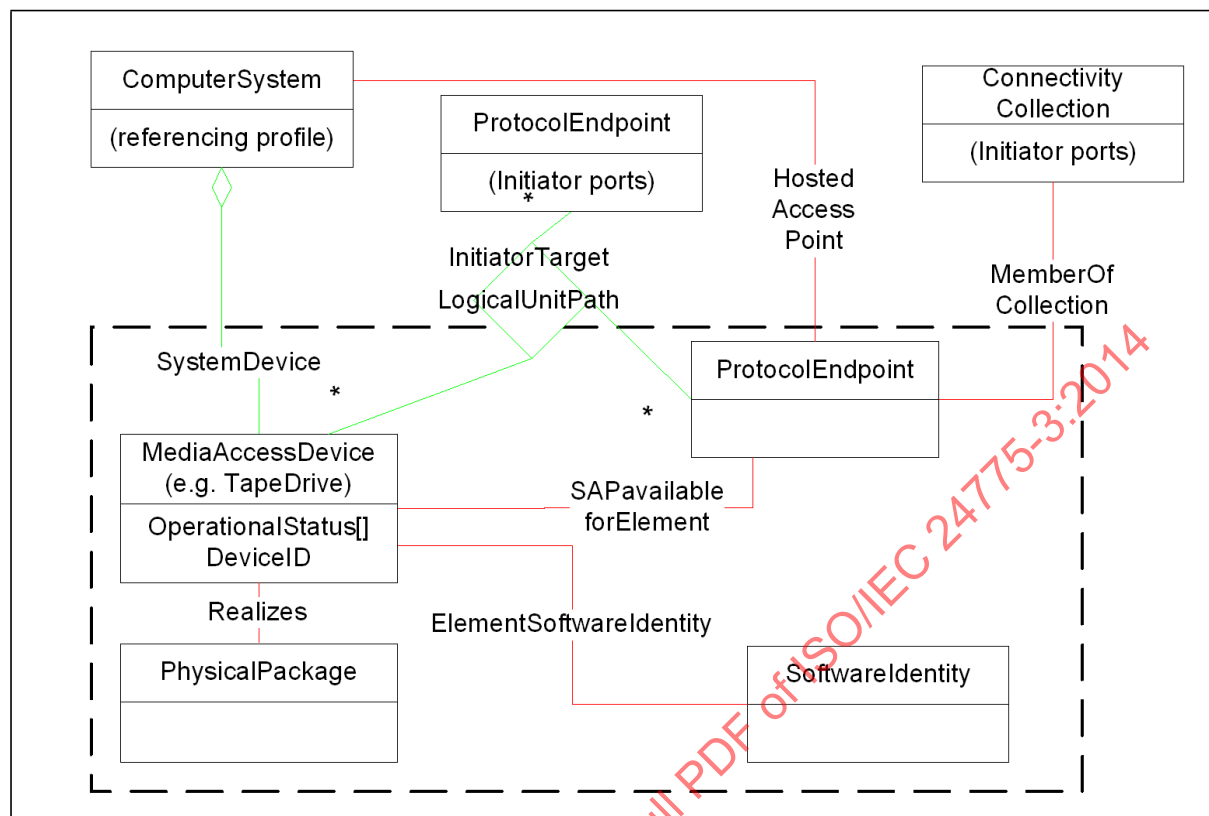


Figure 49 - Media Access Device Class Diagram

37.3 Implementation

37.3.1 Health and Fault Management Consideration

The MediaAccessDevice.OperationalStatus contains the overall status of the disk, summarized in Table 361.

Table 361 - OperationalStatus For MediaAccessDevice

Primary Operational Status	Subsidiary Operational Status	Description
2 "OK"		Media Access Device is enabled.
5 "Predictive Failure"		Media Access Device is functionality nominally but is predicting a failure
6 "Error"		Media Access Device is no longer functioning.
8 "Starting"		Media Access Device is becoming enabled.
9 "Stopping"		Media Access Device is being disabled.
10 "Stopped"		Media Access Device is disabled.

37.3.2 Cascading Considerations

Not defined in this International Standard.

37.3.3 Hot swap insertion or Removal of Drives

Insertion of a drive shall cause an InstCreation indication for the MediaAccessDevice instance. Similarly, hot-swap removal shall cause an InstDelete indication. ProtocolEndpoint, PhysicalPackage, SoftwareInventory, and related associations will also be created and deleted when a drive is inserted or removed, but no indications shall be produced for these other classes.

37.4 Methods

37.4.1 Request State Change

```
uint32 RequestStateChange (
    [In] uint16 RequestedState,
    [Out] CIM_ConcreteJob REF Job,
    [In] datetime TimeoutPeriod)
```

The allowed state changes are indicated by the RequestedStatesSupported property of EnabledLogicalElementCapabilities. A Job shall be returned if the operation takes longer than the TimeoutPeriod. The Requested State of Offline makes a drives extents unavailable to the dependent volume.

The Job may represent a drive rebuild if the RequestedState of the drive is Offline and a failover shall be complete before the offline operation can finish.

37.5 Use Cases

Not defined in this International Standard.

37.6 CIM Elements

37.6.1 General

Table 362 describes the CIM elements for Media Access Device.

Table 362 - CIM Elements for Media Access Device

Element Name	Requirement	Description
37.6.2 CIM_EnabledLogicalElementCapabilities	Mandatory	
37.6.3 CIM_HostedAccessPoint	Optional	ComputerSystem to storage ProtocolEndpoint.
37.6.4 CIM_MediaAccessDevice	Mandatory	Represents a tape or optical drive.
37.6.5 CIM_PhysicalPackage	Optional	The physical aspects of the drive. This is required when modeling physical drives and shall not be implemented for virtual drives in virtual system environments.
37.6.6 CIM_ProtocolEndpoint	Optional	
37.6.7 CIM_Realizes	Mandatory	Associates MediaAccessDevice and PhysicalPackage.
37.6.8 CIM_SAPAvailableForElement	Conditional	Conditional requirement: Support for ProtocolEndpoints. Associates MediaAccessDevice to ProtocolEndpoint.
37.6.9 CIM_SystemDevice	Mandatory	ComputerSystem to MediaAccessDevice.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_MediaAccessDevice	Optional	MediaAccessDevice Creation. See 37.3.3 Hot swap insertion or Removal of Drives.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_MediaAccessDevice	Optional	MediaAccessDevice Removal. See 37.3.3 Hot swap insertion or Removal of Drives.

37.6.2 CIM_EnabledLogicalElementCapabilities

Requirement: Mandatory.

Table 363 describes class CIM_EnabledLogicalElementCapabilities.

Table 363 - SMI Referenced Properties/Methods for CIM_EnabledLogicalElementCapabilities

Properties	Requirement	Description and Notes
RequestedStatesSupported	Mandatory	Possible states that can be requested when using the method RequestStateChange(). If RequestState and RequestStateChange are not implemented then RequestedStatesSupported would indicate none supported.

37.6.3 CIM_HostedAccessPoint

ComputerSystem to storage ProtocolEndpoint.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 364 describes class CIM_HostedAccessPoint.

Table 364 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

37.6.4 CIM_MediaAccessDevice

Represents a tape or optical drive.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 365 describes class CIM_MediaAccessDevice.

Table 365 - SMI Referenced Properties/Methods for CIM_MediaAccessDevice

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
DeviceID	Mandatory	
Name	Mandatory	

Table 365 - SMI Referenced Properties/Methods for CIM_MediaAccessDevice

Properties	Requirement	Description and Notes
OperationalStatus	Mandatory	Shall be 2 5 6 8 10 11 (Okay or Predictive Failure or Error or Starting or Stopping or Stopped).
LocationIndicator	Mandatory	
EnabledState	Mandatory	Possible values: 2 (Enabled - drive is Spun up and online), 3 (Disabled - drive is spun down, and offline), 4 (Shutting down - drive is spinning down), 6 (Enabled but Offline - drive is spun up but offline), 10 (Starting - drive is spinning up).
RequestedState	Optional	Possible RequestedStates: 2 Enabled (Spin up drive if it was spun down and Online the drive if it was offline), 4 (Shut down - spin down drive), 6 (Offline - offline drive).
RequestStateChange()	Conditional	Conditional requirement: Support for online/offline.

37.6.5 CIM_PhysicalPackage

The physical aspects of the drive.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 366 describes class CIM_PhysicalPackage.

Table 366 - SMI Referenced Properties/Methods for CIM_PhysicalPackage

Properties	Requirement	Description and Notes
CreationClassName	Mandatory	
Manufacturer	Mandatory	The name of the organization responsible for producing the PhysicalElement.
Model	Mandatory	The name by which the PhysicalElement is generally known.
Version	Mandatory	The version of the physical element – not necessarily the same as a software/firmware version.
SerialNumber	Mandatory	
PartNumber	Mandatory	

37.6.6 CIM_ProtocolEndpoint

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 367 describes class CIM_ProtocolEndpoint.

Table 367 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	

37.6.7 CIM_Realizes

Associates MediaAccessDevice and PhysicalPackage.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 368 describes class CIM_Realizes.

Table 368 - SMI Referenced Properties/Methods for CIM_Realizes

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

37.6.8 CIM_SAPAvailableForElement

Associates MediaAccessDevice to ProtocolEndpoint.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for ProtocolEndpoints.

Table 369 describes class CIM_SAPAvailableForElement.

Table 369 - SMI Referenced Properties/Methods for CIM_SAPAvailableForElement

Properties	Requirement	Description and Notes
AvailableSAP	Mandatory	
ManagedElement	Mandatory	

37.6.9 CIM_SystemDevice

ComputerSystem to MediaAccessDevice.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 370 describes class CIM_SystemDevice.

Table 370 - SMI Referenced Properties/Methods for CIM_SystemDevice

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	
PartComponent	Mandatory	

EXPERIMENTAL

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

EXPERIMENTAL

38 Storage Enclosure Profile

38.1 Synopsis

Profile Name: Storage Enclosure (Component Profile)

Version: 1.3.0

Organization: SNIA

CIM Schema Version: 2.15.0

Table 371 describes the related profiles for Storage Enclosure.

Table 371 - Related Profiles for Storage Enclosure

Profile Name	Organization	Version	Requirement	Description
Power Supply	SNIA	1.0.1	Optional	
Fan	SNIA	1.0.1	Optional	
Sensors	SNIA	1.0.0	Optional	
Disk Drive Lite	SNIA	1.5.0	Optional	
Media Access Device	SNIA	1.0.0	Optional	
Switch	SNIA	1.5.0	Optional	

The Storage Enclosure Profile describes an enclosure that houses storage components.

38.2 Description

38.2.1 General

The Storage Enclosure Profile describes an enclosure that contains storage elements (e.g., disk or tape drives) and enclosure elements (e.g., fans and power supplies). The logical aspects of the storage and enclosure elements are defined in other profiles. This profile specializes the DMTF Physical Asset Profile adding implementation details for storage enclosures. This profile supports enclosures with a single type of storage component (such as an enclosure of disks) or a mixture of different components.

The following terms apply to this profile:

- **storage elements** are CIM logical classes that relate to storage - CIM_DiskDrive, CIM_ComputerSystem (representing a disk array or switch), etc.
- **enclosure elements** are CIM logical elements that relate to enclosure service and baseboard management - fans, power supplies, sensors, etc.
- **physical elements** are CIM physical classes that map to storage or enclosure elements, and perhaps physical hardware with no logical mapping.

38.2.2 Guidelines related to Referencing Profiles

38.2.2.1 General

The Storage Enclosure Profile is a component profile. The autonomous referencing profile may be Array, Storage Virtualizer, or Host Hardware RAID controller. The following guidelines apply to how this profile is referenced by other profiles.

38.2.2.2 Guideline 1 - enclosure elements dedicated to a single top-level system

If the components of the enclosure are all dedicated to a single top-level System, then the profile defining that system shall be the referencing profile for the enclosure. All components (storage elements, enclosure elements, physical elements) need to be dedicated. For example, if the enclosure is used by a disk array, the CIM_ComputerSystem from the Array profile serves as the scoping instance for all the elements of the enclosure.

Note that the top-level system may be part of an autonomous profile that supports the SNIA Multiple Computer System Profile.

Note that other autonomous profiles may be dedicated as a component of another autonomous profile. For example, a Fibre Channel switch may share an enclosure with, and be dedicated as, a component of an Array.

38.2.2.3 Guideline 2 - enclosure elements shared by multiple top-level systems

If the elements of the enclosure support use by multiple top-level systems, then the referencing profile shall be the base system profile.

Examples include a JBOD array access by multiple servers or multiple switch blades sharing an enclosure.

38.2.2.4 Guideline 3 - enclosure elements need not be scoped by the system as storage elements

CIM requires instantiation of all weak associations whenever the referenced elements are instantiated. For example, every CIM_LogicalDevice instance shall be referenced by a CIM_SystemDevice association. But it is possible to have devices scope to different systems associated to each other by non-weak associations. In particular, when guideline 2 applies, enclosure elements scoped to the enclosure top-level system may be associated to storage elements scoped to a different top-level system. For example, CIM_AssociatedCooling can reference a CIM_Fan scoped to the enclosure system and a CIM_DiskDrive scoped to a server. In another example, CIM_SuppliesPower references a CIM_PowerSupply scoped to an Array within an enclosure and a CIM_ComputerSystem representing a switch.

Figure 50 is an example of two arrays that each have their own enclosure but share cooling. The two array enclosures are contained in an enclosure that provides a fan shared by the array elements.

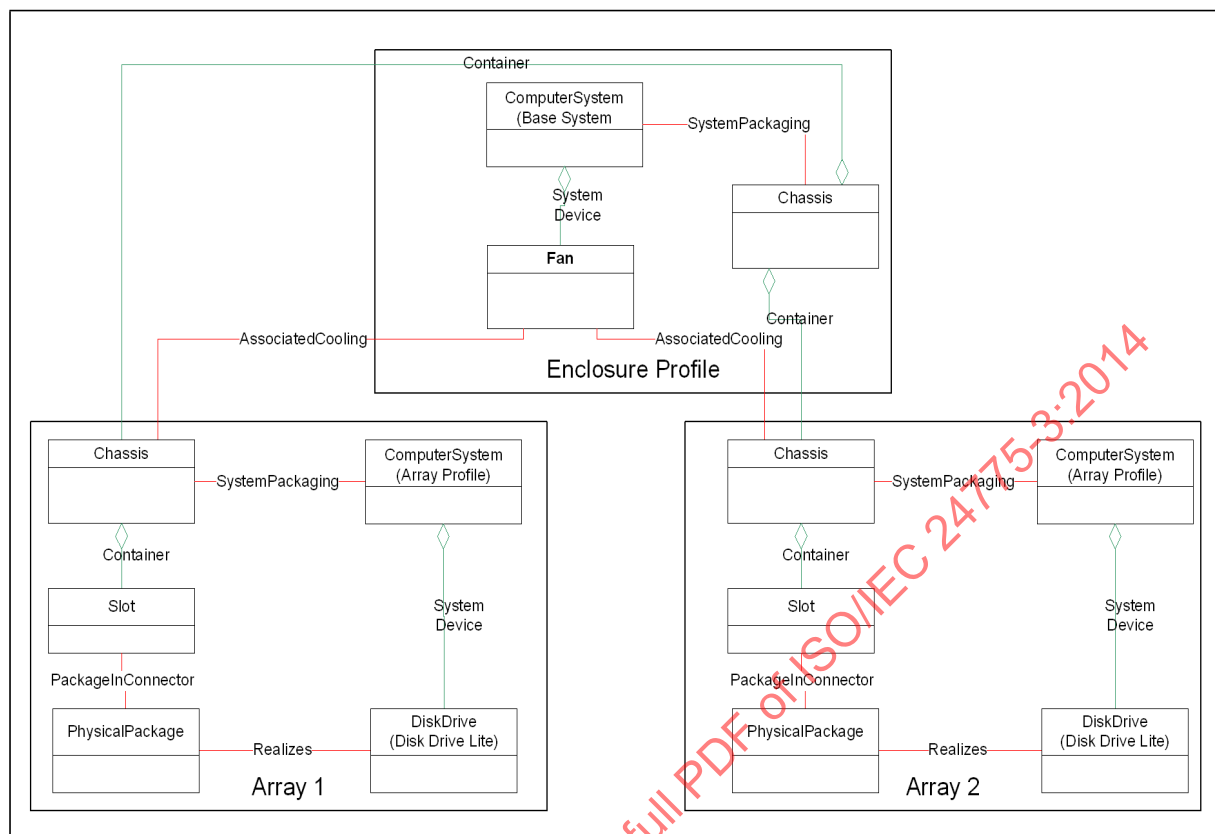


Figure 50 - Enclosure with Two Arrays

38.2.3 Examples of Storage Enclosure Configurations

38.2.3.1 Enclosure Dedicated to a Disk Array

The referencing profile is Array. Disk Drive Lite is a mandatory component profile. The physical model for disks as defined in 38.3.5.3 is mandatory.

38.2.3.2 Enclosure Dedicated to a RAID Host Controller

The referencing profile is the Host Hardware RAID Profile. Support for the Disk Drive Lite Profile is mandatory. The physical model for disks as defined in 38.3.5.3 is mandatory.

38.2.3.3 Enclosure Dedicated to non-RAID Controllers on a Single Server

The referencing profile is the Base System Profile referencing the Storage HBA Profile (or the FC HBA Profile).

38.2.3.4 Enclosure Dedicated to non-RAID Controllers on Multiple Servers

Guideline 2 applies. The referencing profile is the base system profile.

Guideline 3 may apply.

38.2.3.5 FC Switch as a Component of an Array

The Array and FC Switch share an enclosure, but the FC Switch is functionally a sub-component of the array receiving cooling and power from the enclosure. In this configuration, Array is the referencing profile to the Storage Enclosure. Guideline 3 may apply.

38.2.3.6 Enclosure containing multiple FC Switches (Director)

The enclosure is a director class switch which contains one or more switches and other devices including a FCIP Extenders and iSCSI Gateway. The referencing profile is the base system profile. Guideline 2 applies. Guideline 3 may apply.

38.3 Implementation

38.3.1 Health and Fault Management Consideration

Not defined in this International Standard.

38.3.2 Cascading Considerations

Not defined in this International Standard.

38.3.3 Enclosure Elements

38.3.3.1 Power Supplies

A storage enclosure may be modeled with one or more power supplies for device powering.

The CIM_SystemDevice association is used in the Power Supply Profile to connect the power supply to the managed system. The CIM_SuppliesPower association may be used to represent device powering to other enclosure elements of the top-level system as well as logical devices scoped to other systems.

38.3.3.2 Fans

A storage enclosure may be modeled with one or more fans for device cooling.

The CIM_SystemDevice association is used in the Fan Profile to connect the fan to the managed system. The CIM_AssociatedCooling association may be used to represent device powering to other enclosure elements of the top-level system as well as logical devices scoped to other systems.

38.3.3.3 Sensors

A storage enclosure may be modeled with one or more sensors for monitoring such factors as temperature or fan speed.

The CIM_SystemDevice association is used in the Sensors Profile to connect the sensor to the managed system. The CIM_AssociatedSensor association may be used to associate the sensor to other enclosure elements of the top-level system as well as logical devices scoped to other systems.

38.3.4 Storage Elements

38.3.4.1 Considerations for Media Access Devices in a Storage Enclosure

A storage enclosure may contain devices such as disk drives or switches. Each media access device is described by a corresponding device class as described in the corresponding profile. Each device may be associated to a physical bay or slot. The physical model for a disk drive describes a CIM_MediaAccessDevice associated to CIM_PhysicalPackage via CIM_Realizes, and CIM_Slot associated to the CIM_PhysicalPackage via CIM_PackageInConnector. If the implementation also supports hierarchical packaging, the CIM_Slot shall be associated to the CIM_PhysicalPackage realizing the referencing system or an enclosure nested in the system CIM_PhysicalPackage.

38.3.4.2 Disk Drive Considerations

If the implementation also supports the Disk Drive Lite Profile, the individual drives in the storage enclosure shall be described by an instance of CIM_DiskDrive subclassed from CIM_MediaAccessDrive. CIM_PhysicalPackage and CIM_Realizes from the Disk Drive Lite Profile shall provide the instances described in 38.3.4.1.

38.3.4.3 Media Access Devices and the Fan Profile

The Fan Profile describes fans used for device cooling and includes an AssociatedCooling association that references a CIM_ManagedSystemElement. If the implementation supports both the Fan and Disk Drive Lite Profiles, and utilizes the CIM_AssociatedCooling association, the CIM_AssociatedCooling association shall reference an instance of CIM_DiskDrive or an instance of CIM_Chassis.

38.3.4.4 Media Access Devices and the Power Supply Profile

The Power Supply Profile describes power supplies used for device powering and includes a CIM_SuppliesPower association that references a CIM_LogicalDevice. If the implementation supports both the Fan and Disk Drive Lite Profiles, and utilizes the CIM_SuppliesPower association, the CIM_SuppliesPower association shall reference an instance of CIM_DiskDrive or an instance of CIM_Chassis.

38.3.4.5 Configuration Reporting Service

The CIM_ConfigurationReportingService may be used to query for the CIM_MediaAccessDevice or CIM_LogicalPort subclasses supported within the enclosure, the supported total count and the currently installed count. In this way the total number of supported device slots, storage devices or connection ports may be retrieved. See the service method definitions in 38.4.1.

38.3.5 Physical Assets**38.3.5.1 General**

The physical representation of the storage enclosure is mandatory. The core frame of the storage enclosure is described by CIM_Chassis.

38.3.5.2 Physical Package Hierarchy Considerations

A hierarchy of enclosures may be represented. The physical structure of a single enclosure, described by CIM_Chassis, may be associated with a variety of enclosure components and media devices. Any number of CIM_Packages may be used to group physical components. These packages may in turn be associated to one or more CIM_Chassis instances. In this case the CIM_PackageInChassis association shall be used.

38.3.5.3 Disk Drive or Media Access Device

If the implementation models slots within the enclosure, CIM_Slot shall be used to describe the slot. The instance of CIM_PhysicalPackage that describes the physical characteristics of the CIM_DiskDrive instance shall be associated to CIM_Slot by the CIM_PackageInConnector association. If the instance of CIM_Slot is aggregated to an instance of CIM_Chassis, the CIM_ConnectorOnPackage association shall be used. Figure 51 illustrates the model.

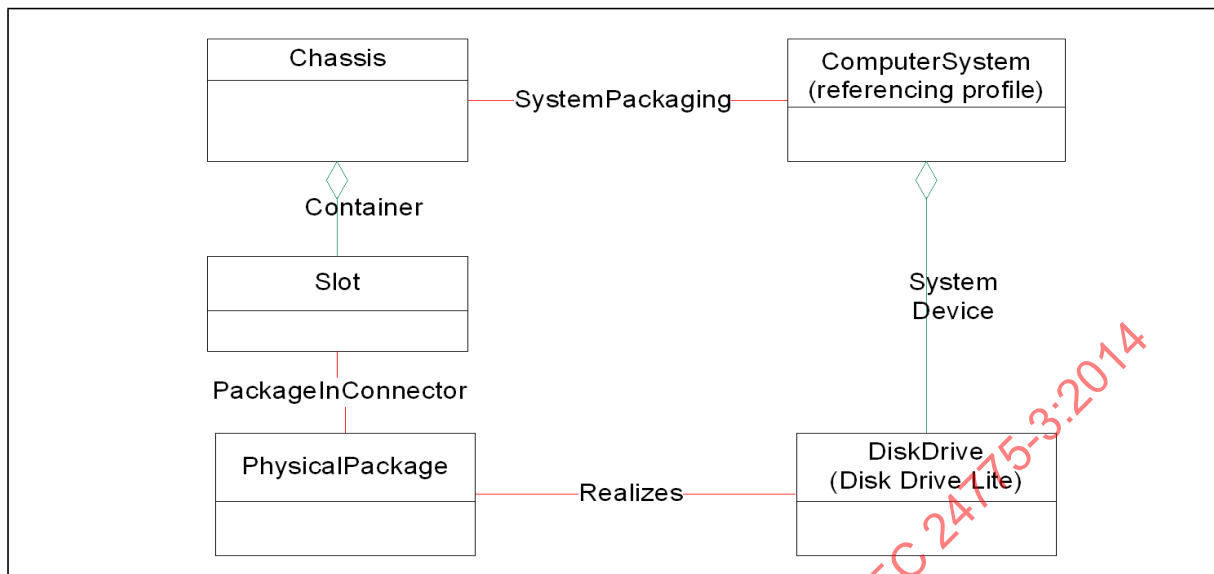


Figure 51 - Model for Disk in Enclosure

38.4 Methods

38.4.1 Extrinsic Methods of the Profile

38.4.1.1 CIM_ConfigurationReportingService GetClassTypes

GetClassTypes is used to query for the supported or currently installed device classes contained in the enclosure such as a CIM_DiskDrive or CIM_SASPort. Reporting of MediaAccessDevice derived classes directly contained within the enclosure (Recursive = False) is mandatory. Reporting of LogicalPort derived classes is optional.

The instrumentation shall support InquiryType parameter values of 2 (Supports) and 3 (Installed).

The instrumentation shall support a Recursive parameter value of false.

The instrumentation shall accept a reference to the top-level ComputerSystem in the Target parameter.

38.4.1.2 CIM_ConfigurationReportingService GetUnitTypes

GetUnitTypes is used to query for the supported or currently installed type of devices contained in the enclosure.

The instrumentation shall support InquiryType parameter values of 2 (Supports) and 3 (Installed).

The instrumentation shall support a Recursive parameter value of false.

The parameter UnitTypes may be set to "Contained", "StorageMediaLocation", "Front Side" or "Back Side". Support of the type "Contained" and "StorageMediaLocation" is mandatory. Support of "Front Side" or "Back Side" is optional. Types "Front Side" or "Back Side" are used to query for the count of the respective LogicalPorts.

38.4.1.3 CIM_ConfigurationReportingService ReportCapacity

ReportCapacity is used after GetClassTypes or GetUnitTypes is issued to find what subclasses and types are available in the enclosure, the ReportCapacity can be used to request the total supported or currently installed storage device slot count or data connection ports for the enclosure.

The instrumentation shall support InquiryType parameter values of 2 (Supports) and 3 (Installed).

The instrumentation shall support a Recursive parameter value of false.

38.4.2 Intrinsic Methods of this Profile

The profile supports read methods and association traversal. Specifically, the list of intrinsic operations supported are as follows:

- GetInstance
- Associators
- AssociatorNames
- References
- ReferenceNames
- EnumerateInstances
- EnumerateInstanceNames

38.5 Use Cases

38.6 CIM Elements

38.6.1 General

Table 372 describes the CIM elements for Storage Enclosure.

Table 372 - CIM Elements for Storage Enclosure

Element Name	Requirement	Description
38.6.2 CIM_ConfigurationReportingService	Mandatory	
38.6.3 CIM_HostedService	Mandatory	Associates the CIM_ConfigurationReportingService to the System in the referencing profile.

38.6.2 CIM_ConfigurationReportingService

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 373 describes class CIM_ConfigurationReportingService.

Table 373 - SMI Referenced Properties/Methods for CIM_ConfigurationReportingService

Properties	Requirement	Description and Notes
ElementName	Mandatory	
Name	Mandatory	
CreationClassName	Mandatory	
GetClassTypes()	Mandatory	
GetUnitTypes()	Mandatory	
ReportCapacity()	Mandatory	

38.6.3 CIM_HostedService

Associates the CIM_ConfigurationReportingService to the System in the referencing profile.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 374 describes class CIM_HostedService.

Table 374 - SMI Referenced Properties/Methods for CIM_HostedService

Properties	Requirement	Description and Notes
Antecedent	Mandatory	The reference to the System.
Dependent	Mandatory	The reference to the Service.

EXPERIMENTAL

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

STABLE**39 Software Subprofile****39.1 Description**

The Software Profile models software or firmware installed on a computer system.

Information on the installed software is given using the SoftwareIdentity class. This is linked to the system using a InstalledSoftwareIdentity association.

Software information may be associated with the “top” level ComputerSystem (if all components are using the same software) or a component ComputerSystem if the software loaded can vary by processor.

Firmware is modeled as SoftwareIdentity. InstalledSoftwareIdentity is used for firmware associated with a System.

Figure 52 contains the instance diagram for the Software Profile.

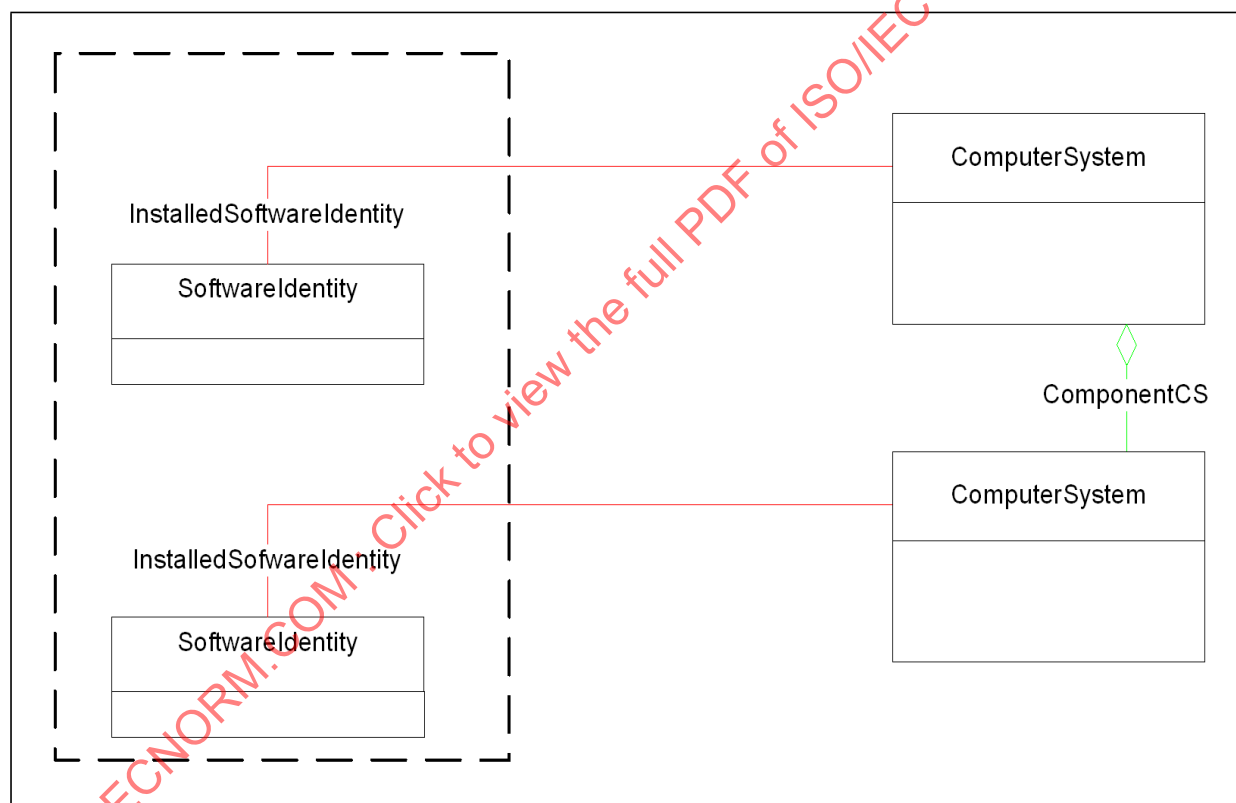


Figure 52 - Software Instance Diagram

39.2 Health and Fault Management Considerations

Not defined in this International Standard.

39.3 Cascading Considerations

Not defined in this International Standard.

39.4 Supported Subprofiles, and Packages

None.

39.5 Methods of the Profile

None.

39.6 Client Considerations and Recipes

None.

39.7 Registered Name and Version

Software version 1.4.0 (Component Profile).

39.8 CIM Elements

39.8.1 General

Table 375 describes the CIM elements for Software.

Table 375 - CIM Elements for Software

Element Name	Requirement	Description
39.8.2 CIM_InstalledSoftwareIdentity	Mandatory	
39.8.3 CIM_SoftwareIdentity	Mandatory	

39.8.2 CIM_InstalledSoftwareIdentity

Create By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 376 describes class CIM_InstalledSoftwareIdentity.

Table 376 - SMI Referenced Properties/Methods for CIM_InstalledSoftwareIdentity

Properties	Flags	Requirement	Description and Notes
System		Mandatory	
InstalledSoftware		Mandatory	

39.8.3 CIM_SoftwareIdentity

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 377 describes class CIM_SoftwareIdentity.

Table 377 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity

Properties	Flags	Requirement	Description and Notes
InstanceID		Mandatory	
VersionString		Mandatory	
Manufacturer		Mandatory	
BuildNumber		Optional	
MajorVersion		Optional	
RevisionNumber		Optional	
MinorVersion		Optional	

STABLE

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

EXPERIMENTAL

40 Software Inventory Profile

40.1 Synopsis

Profile Name: Software Inventory (Component Profile)

Version: 1.0.0

Organization: SNIA

CIM Schema Version: 2.15.0

Table 378 describes the related profiles for Software Inventory.

Table 378 - Related Profiles for Software Inventory

Profile Name	Organization	Version	Requirement	Description
Indication	SNIA	1.5.0	Mandatory	

Specializes: DMTF Software Inventory Profile 1.0.0

Central Class: CIM_SoftwareIdentity

Scoping Class: a CIM_System in a referencing autonomous profile

The Software Inventory Profile models installed and available software and firmware. The SNIA version specializes the DMTF profile in order to add indications.

40.2 Description

40.2.1 General

The Software Inventory Profile models installed and available software and firmware. The SNIA version specializes the DMTF profile in order to add indications.

40.2.2 Relationship to the SMI-S Software Profile

SMI-S defined a similar profile, the Software Subprofile (see Clause 39 "Software Subprofile"). There are several differences between the two profiles.

- The Software Subprofile is limited to modeling software/firmware associated to a system and makes no provision for software/firmware associated to other elements (drives, ports,...)
- The DMTF Software Inventory Profile provides additional functionality:
 - software that is available on the system, but not installed - allowing the ability to model software/firmware that has been downloaded, but not activated.
 - collections of SoftwareIdentity instances
 - locations (such as URLs) associated with SoftwareIdentity instances

Also note that supporting this profile in SMI-S allows us to utilize the DMTF profiles which in turn use the Software Inventory Profile.

Note that although both profiles use InstalledSoftwareIdentity, the semantics are different. In the SMI-S Software Subprofile, InstalledSoftwareIdentity indicates that the software is both available and installed on the system. In the DMTF Software Inventory Profile, InstalledSoftwareIdentity indicates that the software is available (downloaded) on the system, and ElementSoftwareIdentity indicates that the software is active for the referenced element.

Also note that Software Inventory Profile has requirements for version properties beyond those in the SNIA Software Subprofile.

40.3 Implementation

40.3.1 General

See DSP1023, DMTF Software Inventory Profile.

40.3.2 Software Installation and Update

The CIM interface for Software Updates is described in the DMTF Software Update Profile (DSP1025). As a side effect of installation or updates, the inventory of software identities modeled in this profile is modified. This specialization adds indication filters:

- InstCreation of SoftwareIdentity represents a newly available software element (or new version);
- InstDeletion of SoftwareIdentity represents the deletion of an inactive SoftwareIdentity;
- InstAlert with a Standard Message is used when a software (or firmware) version is updated “in-place” without installing a separate software/firmware package;
- InstModification of ElementSoftwareIdentity.ElementSoftwareStatus (see 7.4.1.1 in DSP1023, DMTF Software Inventory Profile).

40.3.3 Health and Fault Management Consideration

None.

40.3.4 Cascading Considerations

None.

40.4 Methods

See DSP1023, DMTF Software Inventory Profile.

40.5 Use Cases

See DSP1023, DMTF Software Inventory Profile.

40.6 CIM Elements

40.6.1 General

Table 379 describes the CIM elements for Software Inventory.

Table 379 - CIM Elements for Software Inventory

Element Name	Requirement	Description
40.6.2 CIM_ElementSoftwareIdentity	Optional	
40.6.3 CIM_HostedAccessPoint	Optional	
40.6.4 CIM_HostedCollection	Conditional	Conditional requirement: Support for collection of SoftwareIdentity instances.
40.6.5 CIM_InstalledSoftwareIdentity	Optional	
40.6.6 CIM_MemberOfCollection	Conditional	Conditional requirement: Support for collection of SoftwareIdentity instances.
40.6.7 CIM_OrderedComponent	Optional	
40.6.8 CIM_OrderedDependency	Optional	
40.6.9 CIM_SAPAvailableForElement	Conditional	Conditional requirement: Support for SoftwareIdentityResource instances.

Table 379 - CIM Elements for Software Inventory

Element Name	Requirement	Description
40.6.10 CIM_SoftwareIdentity	Mandatory	
40.6.11 CIM_SoftwareIdentityResource	Optional	
40.6.12 CIM_SystemSpecificCollection	Optional	
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_SoftwareIdentity	Mandatory	Creation of a SoftwareIdentity. See 40.3.2 Software Installation and Update.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_SoftwareIdentity	Mandatory	Delete of a SoftwareIdentity. See 40.3.2 Software Installation and Update.
SELECT * FROM CIM_AlertIndication WHERE OwningEntity=SNIA and MessageID=\Core1\	Mandatory	In-place update of Software (or Firmware). See 40.3.2 Software Installation and Update.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_SoftwareIdentity AND SourceInstance.CIM_SoftwareIdentity::ElementSoftwareStatus <> PreviousInstance.CIM_SoftwareIdentity::ElementSoftwareStatus	Optional	CQL - Change in ElementSoftwareStatus property of SoftwareIdentity. See 40.3.2 Software Installation and Update.

40.6.2 CIM_ElementSoftwareIdentity

CIM_ElementSoftwareIdentity is used to associate an instance of CIM_ManagedElement and an instance of CIM_SoftwareIdentity when the instance of CIM_ManagedElement is instrumented.

Requirement: Optional.

Table 380 describes class CIM_ElementSoftwareIdentity.

Table 380 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity

Properties	Requirement	Description and Notes
ElementSoftwareStatus	Mandatory	
Antecedent	Mandatory	
Dependent	Mandatory	

40.6.3 CIM_HostedAccessPoint

CIM_HostedAccessPoint is used to associate CIM_System and CIM_SoftwareIdentityResource when an instance of CIM_SoftwareIdentityResource is instrumented.

Requirement: Optional.

Table 381 describes class CIM_HostedAccessPoint.

Table 381 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

40.6.4 CIM_HostedCollection

CIM_HostedCollection is used to associate CIM_System and CIM_SystemSpecificCollection. CIM_HostedCollection is conditional and shall be implemented when an instance of CIM_SystemSpecificCollection is instrumented.

Requirement: Support for collection of SoftwareIdentity instances.

Table 382 describes class CIM_HostedCollection.

Table 382 - SMI Referenced Properties/Methods for CIM_HostedCollection

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

40.6.5 CIM_InstalledSoftwareIdentity

CIM_InstalledSoftwareIdentity is used to associate an instance of CIM_System and an instance of CIM_SoftwareIdentity. CIM_InstalledSoftwareIdentity is conditional and shall be implemented when Installed Software is modeled.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 383 describes class CIM_InstalledSoftwareIdentity.

Table 383 - SMI Referenced Properties/Methods for CIM_InstalledSoftwareIdentity

Properties	Requirement	Description and Notes
System	Mandatory	
InstalledSoftware	Mandatory	

40.6.6 CIM_MemberOfCollection

CIM_MemberOfCollection is used to associate an instance of CIM_SystemSpecificCollection and an instance of CIM_SoftwareIdentity. CIM_MemberOfCollection is conditional and shall be implemented when an instance of CIM_SystemSpecificCollection is instrumented.

Requirement: Support for collection of SoftwareIdentity instances.

Table 384 describes class CIM_MemberOfCollection.

Table 384 - SMI Referenced Properties/Methods for CIM_MemberOfCollection

Properties	Requirement	Description and Notes
Collection	Mandatory	
Member	Mandatory	

40.6.7 CIM_OrderedComponent

CIM_OrderedComponent is used to associate an instance of CIM_SoftwareIdentity that represents a Software Bundle and an instance of CIM_SoftwareIdentity that represents one of the discrete software images contained in the Software Bundle.

Requirement: Optional.

Table 385 describes class CIM_OrderedComponent.

Table 385 - SMI Referenced Properties/Methods for CIM_OrderedComponent

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	
PartComponent	Mandatory	

40.6.8 CIM_OrderedDependency

CIM_OrderedDependency is used to associate an instance of CIM_SoftwareIdentity that represents an Installation Dependency and an instance of CIM_SoftwareIdentity for which the Installation Dependencies are represented.

Requirement: Optional.

Table 386 describes class CIM_OrderedDependency.

Table 386 - SMI Referenced Properties/Methods for CIM_OrderedDependency

Properties	Requirement	Description and Notes
AssignedSequence	Mandatory	
Antecedent	Mandatory	
Dependent	Mandatory	

40.6.9 CIM_SAPAvailableForElement

CIM_SAPAvailableForElement is used to associate CIM_SoftwareIdentityResource and CIM_SoftwareIdentity. CIM_SAPAvailableForElement is conditional and shall be implemented when the location information of CIM_SoftwareIdentity is represented.

Requirement: Support for SoftwareIdentityResource instances.

Table 387 describes class CIM_SAPAvailableForElement.

Table 387 - SMI Referenced Properties/Methods for CIM_SAPAvailableForElement

Properties	Requirement	Description and Notes
AvailableSAP	Mandatory	
ManagedElement	Mandatory	

40.6.10 CIM_SoftwareIdentity

CIM_SoftwareIdentity is used to represent either Installed Software or Available Software.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 388 describes class CIM_SoftwareIdentity.

Table 388 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
IsEntity	Mandatory	
VersionString	Optional	
BuildNumber	Optional	
MajorVersion	Conditional	Conditional requirement: No Support for SoftwareIdentity.VersionString.
MinorVersion	Conditional	Conditional requirement: No Support for SoftwareIdentity.VersionString.
RevisionNumber	Conditional	Conditional requirement: No Support for SoftwareIdentity.VersionString.
TargetOSTypes	Optional	

40.6.11 CIM_SoftwareIdentityResource

CIM_SoftwareIdentityResource is used to represent the location of a Software Identity, which could be used as input to the software installation service.

Requirement: Optional.

Table 389 describes class CIM_SoftwareIdentityResource.

Table 389 - SMI Referenced Properties/Methods for CIM_SoftwareIdentityResource

Properties	Requirement	Description and Notes
SystemCreationClassName	Mandatory	
SystemName	Mandatory	
CreationClassName	Mandatory	
Name	Mandatory	
InfoFormat	Mandatory	
AccessInfo	Mandatory	
ResourceType	Optional	

40.6.12 CIM_SystemSpecificCollection

CIM_SystemSpecificCollection is used to represent a collection of Available Software.

Requirement: Optional.

Table 390 describes class CIM_SystemSpecificCollection.

Table 390 - SMI Referenced Properties/Methods for CIM_SystemSpecificCollection

Properties	Requirement	Description and Notes
InstanceID	Mandatory	
ElementName	Mandatory	

EXPERIMENTAL

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

STABLE

41 Server Profile

41.1 Description

41.1.1 Model Overview

A CIM Server is anything that supports the CIM-XML protocol or other WBEM protocols. The Server Profile is mandatory for all compliant Information Technology – Storage Management servers.

The object manager part of the model, shown in Figure 53, defines the capabilities of a CIM object manager based on the communication mechanisms that it supports.

The namespace model of the Server Profile describes the namespaces managed by the object manager and the type information contained within the namespace. The main information provided in the namespace part of the model is the namespace itself and its association to the ObjectManager.

The InteropNamespace refers to the first namespace found in the InteropSchemaNamespace attribute of the SLP Template.

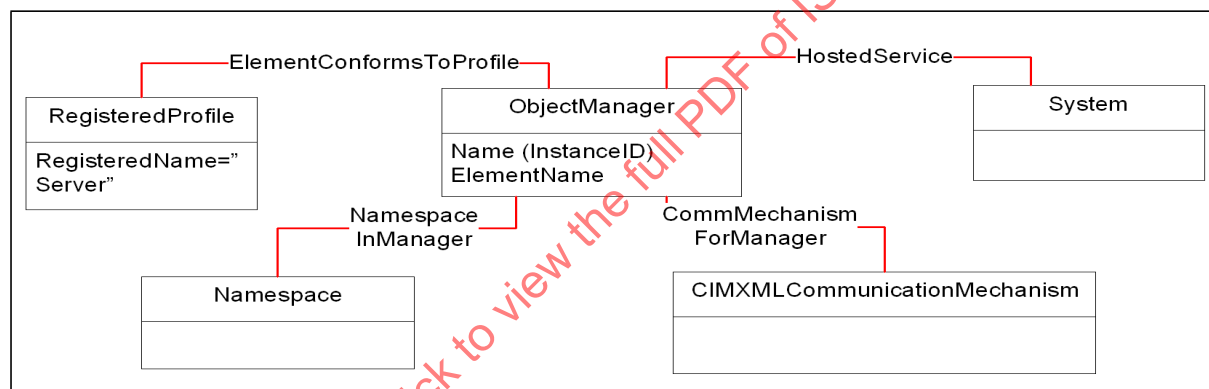


Figure 53 - Server Model

A Server is modeled as a System with a HostedService association to an ObjectManager. The ObjectManager is subclassed from Service. Implementations shall support an ElementConformsToProfile association referencing the RegisteredProfile for the Server Profile and referencing the ObjectManager (rather than CIM_System as is common in other profiles).

It is mandatory that all namespaces supported by the Server be identified (the Namespace class) and associated to the ObjectManager via the NamespaceInManager association.

The communication protocols supported by the ObjectManager should also be identified. Specifically, the CIMXMLCommunicationMechanism shall be present for standard communication support for clients. This class is associated to the ObjectManager via the CommMechanismForManager association.

The Profile Registration Profile describes the set of classes and associations deal with profiles supported by the ObjectManager. The Profile Registration Profile is required by the Server Profile.

41.1.2 Use of model fields to Populate the SLP template

The data used to populate the SLP template for advertising Information Technology – Storage Management profiles is found in the CIM Server profile. The SLP template fields are populated as follows:

template-url-syntax: =string

The following quotation is from the "WBEM SLP Template v1.0.0.

<http://www.dmtf.org/sites/default/files/standards/documents/wbem.1.0.en>

"The template-url-syntax MUST be the WBEM URI Mapping of the location of one service access point offered by the WBEM Server over TCP transport. This attribute must provide sufficient addressing information so that the WBEM Server can be addressed directly using the URL.

The WBEM URI Mapping is defined in the WBEM URI Mapping Specification 1.0.0 (DSP0207). Example: (template-url-syntax=https://localhost:5989)"

service-hi-name: ObjectManager.ElementName

service-hi-description: ObjectManager.Description

service-id: ObjectManager.Name

CommunicationMechanism:

ObjectManagerCommunicationMechanism.CommunicationMechanism

OtherCommunicationMechanism:

ObjectManagerCommunicationMechanism.OtherCommunicationMechanism

InteropSchemaNamespace: Namespace.Name for the InteropNamespace

ProtocolVersion: ObjectManagerCommunicationMechanism.Version

MultipleOperationsSupported:

ObjectManagerCommunicationMechanism.MultipleOperationsSupported

AuthenticationMechanismSupported:

ObjectManagerCommunicationMechanism.AuthenticationMechanismsSupported

OtherAuthenticationDescription:

ObjectManagerCommunicationMechanism.AuthenticationMechanismDescriptions

Namespace: Namespace.Name for each Namespace instance supported

Classinfo: Namespace.Classinfo for each Namespace instance

RegisteredProfilesSupported:

A list of profiles supported by the CIM providers running in this CIM Server. Each entry in this list is separated by a comma and consists of two or three sub-fields, separated by colons. If an entry refers to a supported profile defined in a RegisteredProfile (and not RegisteredSubProfile) instance, the format shall be

Organization:Name

where organization is the name of the organization that defined the profile (e.g., SNIA or DMTF) and Name is the name of the profile. Note that this first format applies to autonomous or component profiles defined using RegisteredProfile. If an entry refers to a supported subprofile defined in a RegisteredSubProfile instance, the format shall be

Organization:Name:Subprofile-Name

where organization is the name of the organization that defined the profile (e.g., SNIA or DMTF), Name is the name of the profile, and Subprofile-Name is the name of the subprofile.

For either format, Organization shall be identical to the RegisteredOrganization attribute in the appropriate RegisteredProfile instance. For the first format, Name shall be identical to the RegisteredName attribute in the appropriate RegisteredProfile instance. For the second format:

- Subprofile-Name shall be identical to the RegisteredName attribute in the appropriate RegisteredSubProfile instance;
- Name shall be identical to the RegisteredName attribute in the RegisteredProfile referenced by the RegisteredSubProfile.

Implementations are required to include an entry for each supported autonomous profile. Implementations are required to include an entry for a component profile if the component profile definition in this International Standard states that the component profile shall be advertised via SLP. It is recommended that other subprofiles and component profiles be excluded from this list to minimize the size of the SLP template.

41.1.3 Support for Indications

An implementation of the Server Profile may optionally support an indication reporting that the Object Manager was started.

41.1.4 Security Background

See *Information Technology – Storage Management, Part 2 Common Architecture*, Security (13)

41.2 Health and Fault Management

Not defined in this International Standard.

41.3 Cascading Considerations

Not defined in this International Standard.

41.4 Supported Subprofiles and Packages

Table 391 describes the supported profiles for Server.

Table 391 - Supported Profiles for Server

Profile Name	Organization	Version	Requirement	Description
Object Manager Adapter	SNIA	1.3.0	Optional	
Indication	SNIA	1.5.0	Optional	
Experimental Indication	SNIA	1.5.0	Optional	
Profile Registration	SNIA	1.5.0	Mandatory	
Launch In Context	DMTF	1.0.0	Optional	Experimental. See DSP1102, version 1.0.0

41.5 Methods of the Profile

The implementation may support ObjectManager.StopService. If implemented, this method shall shut down the CIM object manager. The method returns an integer value of 0 if the service was successfully stopped, 1 if the request is not supported, and any other number to indicate an error.

41.6 Client Considerations and Recipes – Segregate a SAN Device Type

```
// DESCRIPTION
// A management application wishes to manage a particular type of SAN
// device, but not other devices. So the management application needs to
// isolate the particular CIM Servers that support the type of device it
// wants to manage.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1.Assume CIM Servers have advertised their services (SrvReg)
// 2.Assume there are one or more Directory Agents in the subnet
```

```
// 3.Assume no security on SLP discovery
// 4.#DirectoryList[] is an array of directory URLs
// 5.#DirectoryEntries [] is an array of directory entry Structures.
// The structure matches the "wbem" SLP Template (see 'Standard
// WBEM Service Type Templates).
// 6.Assume that the device is #DesiredProfile and the device is an
// SMI-S device (a SNIA defined profile)

// Step 1: Set the Previous Responders List to the Null String.
#PRList = ""

// Step 2: Multicast a Service Request for a Directory Server Service.
// This is to find Directory Agents in the subnet.
//
SrvRqst (
    #PRList,          // The Previous Responders list
    service:directory-agent // Service type
    "DEFAULT",        // The scope
    NULL,             // The predicate
    NULL)             // SLP SPI (security token)

// Step 3: Listen for Response from Directory Agent(s)
#DirectoryList[] = DAAdvert (
    BootTimestamp, // Time of last reboot of DA
    URL,           // The URL of the DA
    ScopeList, // The scopes supported by the DA
    AttrList, // The DA Attributes
    SLP SPI List, // SLP SPI (SPIs the DA can verify)
    AuthenticationBlock)
// Iterate on Steps 2 & 3, until a response has been received or the
// client
// has reached a UA configured CONFIG_RETRY_MAX seconds.

// Step 4: Unicast a Service Request to each of the DAs specifying a
// query predicate to select CIM Servers that support SNIA
// #DesiredDevice profiles and listen for responses.
for #j in #DirectoryList[]
{
    SrvRqst (
        #DAPRList,          // The Previous Responders list
        "service:wbem",     // Service type
        "DEFAULT",          // The scope
        "RegisteredProfilesSupported=SNIA:"+#DesiredProfile+"*",
                                // The predicate
        NULL)               // SLP SPI (security token)
    #ServiceList [#j] = SrvRply (
        Count,              // count of URLs
        #SAPRList[])
}
```

```

// Step 5: Next retrieve the attributes of each advertisement
For #i in #ServiceList[] // for each url in list
{
    AttrRqst (
        #SAPRList,          // The Previous Responders list
        #ServiceList[#i ], // a url from #ServiceList[]
        "DEFAULT", // The scope
        NULL, // Tag list. NULL means return all
                // attributes
        NULL) // SLP SPI (security token)
    #DirectoryEntries [#i] = AttrRply (#attr-list)
}

// Step 7: Correlate the responses to the Service Request on unique
// "service-id" to determine unique CIM Servers. The client will get
// multiple responses (one for each access point) for each CIM
// Server. At this point, the client has a list of CIM Servers that
// claim to support SNIA #DesiredProfile profiles.

```

41.7 Registered Name and Version

Server version 1.5.0 (Autonomous Profile)

41.8 CIM Elements

41.8.1 General

Table 392 describes the CIM elements for Server.

Table 392 - CIM Elements for Server

Element Name	Requirement	Description
41.8.2 CIM_CIMXMLCommunicationMechanism	Mandatory	
41.8.3 CIM_CommMechanismForManager	Mandatory	This associates the ObjectManager and the communication classes it supports.
41.8.4 CIM_HostedAccessPoint	Mandatory	This associates the communication mechanisms with the hosting System.
41.8.5 CIM_HostedService	Mandatory	Connects the ObjectManager to the System that is hosting the ObjectManager.
41.8.6 CIM_Namespace	Mandatory	This is a namespace within the Object Manager.
41.8.7 CIM_NamespaceInManager	Mandatory	This associates the namespace to the ObjectManager.
41.8.8 CIM_ObjectManager	Mandatory	This is the Object Manager service of the CIM Server. Associated to RegisteredProfile.
41.8.9 CIM_ObjectManagerCommunicationMechanism	Optional	Instantiate to show optional support for WS-Man.

Table 392 - CIM Elements for Server

Element Name	Requirement	Description
41.8.10 CIM_System	Mandatory	The System that is hosting the Object Manager (CIM Server).
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ObjectManager AND SourceInstance.Started <> PreviousInstance.Started	Optional	Deprecated WQL - Start of object manager. See 41.1.3 Support for Indications.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ObjectManager AND SourceInstance.CIM_ObjectManager::Started <> PreviousInstance.CIM_ObjectManager::Started	Optional	CQL - Start of object manager. See 41.1.3 Support for Indications.

41.8.2 CIM_CIMXMLCommunicationMechanism

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 393 describes class CIM_CIMXMLCommunicationMechanism.

Table 393 - SMI Referenced Properties/Methods for CIM_CIMXMLCommunicationMechanism

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	
ElementName		Mandatory	
CommunicationMechanism		Mandatory	Shall be 2 (CIM-XML).
Version		Mandatory	The 'CIM Operations over HTTP' version supported by the server.
CIMValidated		Mandatory	
FunctionalProfilesSupported		Mandatory	
MultipleOperationsSupported		Mandatory	
AuthenticationMechanismsSupported		Mandatory	
OperationalStatus		Mandatory	Should be 0 (Unknown), 2 (Okay), or 10 (Stopped).
StatusDescriptions		Optional	
FunctionalProfileDescriptions		Optional	

41.8.3 CIM_CommMechanismForManager

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 394 describes class CIM_CommMechanismForManager.

Table 394 - SMI Referenced Properties/Methods for CIM_CommMechanismForManager

Properties	Flags	Requirement	Description and Notes
Dependent		Mandatory	
Antecedent		Mandatory	

41.8.4 CIM_HostedAccessPoint

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 395 describes class CIM_HostedAccessPoint.

Table 395 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint

Properties	Flags	Requirement	Description and Notes
Dependent		Mandatory	
Antecedent		Mandatory	

41.8.5 CIM_HostedService

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 396 describes class CIM_HostedService.

Table 396 - SMI Referenced Properties/Methods for CIM_HostedService

Properties	Flags	Requirement	Description and Notes
Dependent		Mandatory	
Antecedent		Mandatory	

41.8.6 CIM_Namespace

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 397 describes class CIM_Namespace.

Table 397 - SMI Referenced Properties/Methods for CIM_Namespace

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
ObjectManagerCreationClassName		Mandatory	
ObjectManagerName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	
ClassType		Mandatory	Should be 2 (CIM) for either CIM_ and SNIA_ classes.
DescriptionOfClassType		Conditional	Conditional requirement: CIM_Namespace.Namespace having value 1 (Other).
ClassInfo		Optional	Deprecated. Deprecated in the MOF, but required for 1.0 compatibility. Required only if the CIMOM is hosting profiles that were part of SMI-S 1.0.x.
DescriptionOfClassInfo		Optional	Deprecated. Deprecated in the MOF, but mandatory for 1.0 compatibility. Mandatory if ClassInfo is set to 'Other' and if the CIMOM is hosting profiles that were part of SMI-S 1.0.x.

41.8.7 CIM_NamespaceInManager

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 398 describes class CIM_NamespaceInManager.

Table 398 - SMI Referenced Properties/Methods for CIM_NamespaceInManager

Properties	Flags	Requirement	Description and Notes
Dependent		Mandatory	
Antecedent		Mandatory	

41.8.8 CIM_ObjectManager

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Shall be associated to RegisteredProfile using ElementConformsToProfile association. The RegisteredProfile instance shall have RegisteredName set to 'Server', *Information Technology – Storage Management, Part 1 Overview* RegisteredOrganization set to 'SNIA', and RegisteredVersion set to '1.5.0'.

Table 399 describes class CIM_ObjectManager.

Table 399 - SMI Referenced Properties/Methods for CIM_ObjectManager

Properties	Flags	Requirement	Description and Notes
Name		Mandatory	
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
ElementName		Mandatory	The name (make and model) of the server for human interfaces. For example, "ACME CIM Server".
Description		Mandatory	The name (make and model) and version of the server for human interfaces. For example, "ACME CIM Server version 2.2".
OperationalStatus		Mandatory	Should be 0 (Unknown), 2 (Okay), or 10 (Stopped).
Started		Mandatory	
StopService()		Optional	This method shall shut down the CIM object manager. See this standard. The implementation may support ObjectManager.StopService. If implemented, this method shall shut down the CIM object manager. The method returns an integer value of 0 if the service was successfully stopped, 1 if the request is not supported, and any other number to indicate an error.

41.8.9 CIM_ObjectManagerCommunicationMechanism

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 400 describes class CIM_ObjectManagerCommunicationMechanism.

Table 400 - SMI Referenced Properties/Methods for CIM_ObjectManagerCommunicationMechanism

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	
ElementName		Mandatory	
CommunicationMechanism		Mandatory	Shall be 4 (WS Management).
FunctionalProfilesSupported		Mandatory	
MultipleOperationsSupported		Mandatory	

**Table 400 - SMI Referenced Properties/Methods for
CIM_ObjectManagerCommunicationMechanism**

Properties	Flags	Requirement	Description and Notes
AuthenticationMechanismsSupported		Mandatory	
OperationalStatus		Mandatory	Should be 0 (Unknown), 2 (Okay), or 10 (Stopped).
StatusDescriptions		Optional	
FunctionalProfileDescriptions		Optional	

41.8.10CIM_System

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 401 describes class CIM_System.

Table 401 - SMI Referenced Properties/Methods for CIM_System

Properties	Flags	Requirement	Description and Notes
CreationClassName		Mandatory	
NameFormat		Mandatory	Shall be either 'IP', 'WWN', or 'Other'.
Name		Mandatory	If NameFormat is 'IP', Name shall be a valid ipv4, ipv6, or fully qualified domain name. If NameFormat, is 'WWN', Name shall be formatted as 16 unseparated upper case hex digits.
Description		Mandatory	
ElementName		Mandatory	
OperationalStatus		Mandatory	Should be 0 (Unknown), 2 (Okay), or 10 (Stopped).

STABLE

STABLE

42 Profile Registration Profile

42.1 Synopsis

Profile Name: Profile Registration

Version: 1.0.0

Organization: SNIA

CIM Schema Version: 2.12.0

Specializes: DMTF Profile Registration 1.0.0

No included profiles are defined in this International Standard.

Profile Registration Profile models the profiles registered in the object manager and the associations between registration classes and the domain classes implementing the profile.

42.2 Description

The SNIA Profile Registration Profile specializes the DMTF Profile Registration Profile adding the following classes:

- CIM_RegisteredSubProfile (subclass of CIM_RegisteredProfile)
- CIM_SubProfileRequiresProfile (subclass of CIM_ReferencedProfile)
- CIM_SoftwareIdentity
- CIM_ElementSoftwareIdentity
- CIM_Product
- CIM_ProductSoftwareComponent

42.3 Implementation

42.3.1 General

In DMTF profiles, the term 'component profile' is used similarly to the way 'subprofile' was used in SMI-S 1.0.x and 1.1.x, and the term 'autonomous profile' is used similarly to the way 'profile' was used in SMI-S 1.0.x and 1.1.x. SNIA implementations may use the SNIA 1.0.x/1.1.x approach with the RegisteredSubProfile and SubProfileRequiresProfile subclasses or the DMTF approach using Registered Profile for component profiles and ReferencedProfile. Figure 54 shows the Profile Registration Model.

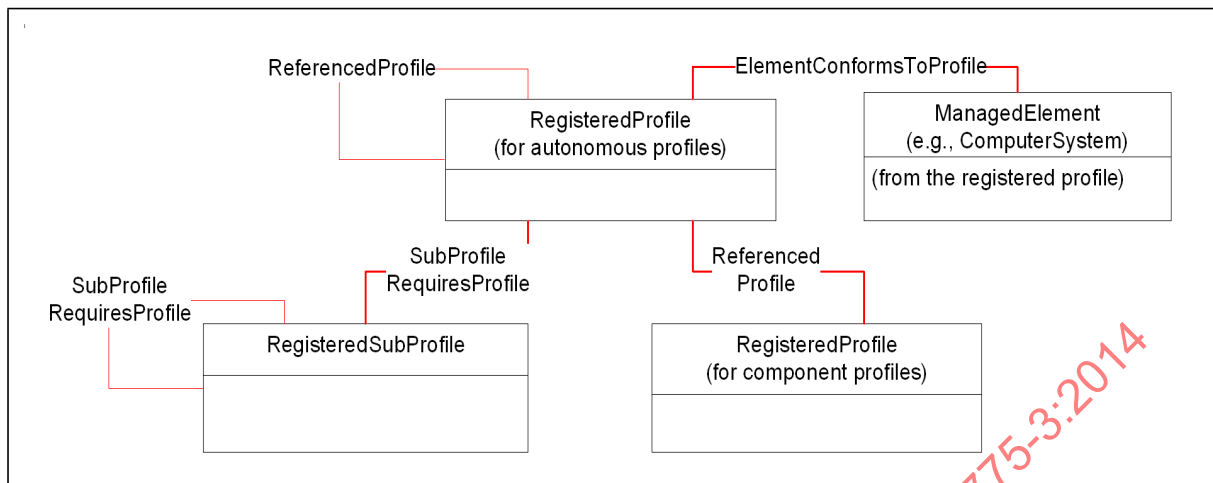


Figure 54 - Profile Registration Model

SMI-S clients should use the superclasses (RegisteredProfile and ReferencedProfile) in CIM operations to assure that implementations conforming to either SMI-S or DMTF profiles are discovered. ReferencedProfile associates two instances of RegisteredProfile. The DMTF Profile Registration Profile describes how the Antecedent and Dependent references should be used when one profile includes another in its supported/referenced profile list. Implementations are inconsistent in the use of these references and clients should be prepared for either approach. One technique to achieve this would be to specify NULL for Role and RemoteRole in Associator or AssociatorName operations.

The Scoping Class methodology defined in the DMTF Profile Registration Profile shall be implemented. The Central Class methodology may be implemented.

For each Profile instance, the supported component profiles should be identified via the SubprofileRequiresProfile or ReferencedProfile association. Subprofiles are modeled using RegisteredSubProfile (or ReferencedProfile).

Instances of RegisteredProfile, RegisteredSubProfile, SubProfileRequiresProfile, and ReferencedProfile are in the Interop namespace. The ManagedElement is in the implementation namespace.

For implementations conforming to SMI-S 1.0.x or 1.1.x, all RegisteredProfile/RegisteredSubProfile instances associated via SubProfileRequiresProfile shall have the version number of the SMI-S standard for the RegisteredVersion property. For implementations conforming to SMI-S 1.2.x or later, the RegisteredVersion property of associated RegisteredProfile/RegisteredSubProfile instances may have different values. These values shall be the same as those published in the *Registered Name and Version* subclause of the profiles. The version of the standard shall be expressed using the SMI-S RegisteredProfile (see 42.3.4 "The SMI-S Registered Profile").

RegisteredProfile instances are required for all SMI-S profiles, including those named as Subprofiles or Packages.

42.3.2 ElementConformsToProfile Association

In addition, the ElementConformsToProfile association ties the RegisteredProfile for SMI-S autonomous profiles to scoping managed elements (typically ComputerSystems). These are the "top-level" objects defined in SMI-S autonomous profiles. Implementations shall support an ElementConformsToProfile association from at least one RegisteredProfile instance to the scoping instance.

A single ManagedElement may have zero or more ElementConformsToProfile associations to RegisteredProfiles. Regardless of the number of associated RegisteredProfiles, the ManagedElement represents one set of resources. So for example, consider a ManagedElement that is a System that supports both the Array and Storage Virtualizer profiles. If one asks for the total amount of mapped capacity, the answer applies to both Array and Virtualizer and is not additive. See Clause B.6 "Rules for Combining (Autonomous) Profiles" in *Information Technology – Storage Management, Part 2 Common Architecture*.

42.3.3 Associations between Autonomous and Component Profile

The DMTF Profile Registration Profile requires the RegisteredProfile instances representing a profile and its supported profiles be associated via ReferencedProfile (which may be subclassed as SubProfileRequiresProfile). SMI-S has the additional requirement, that all supported profiles (whether supported directly or indirectly), are associated directly to the "top-level" autonomous profile.

For example, as shown in Figure 55, the Array Profile supports the Disk Sparing Subprofile which supports the Job Control. SMI-S requires both of these component profiles to be directly attached to the Array Profile instance, even though Job Control is actually a component profile of Disk Sparing. DMTF Profile Registration Profile also requires a ReferencedProfile association between the RegisteredProfiles for Disk Sparing and Job Control.

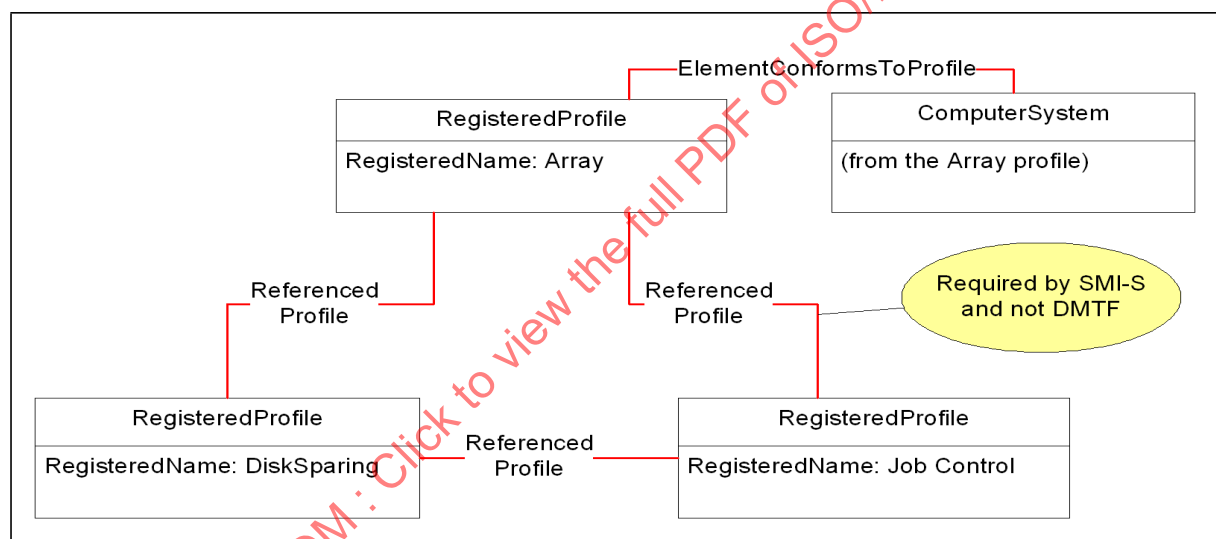


Figure 55 - Associations between RegisteredProfile instances

Each RegisteredProfile instance referenced by ElementConformsToProfile may have a set of supported profiles with RegisteredProfile instances associated using ReferencedProfile or SubProfileRequiresProfile. Typically the RegisteredProfile associated via ElementConformsToProfile is for an autonomous profile and the supported profiles are component profiles. If there are multiple ElementConformsToProfile associations between a single RegisteredProfile instance and multiple domain instances, the referenced domain implementations shall support all the profiles supported by the RegisteredProfile.

42.3.4 The SMI-S Registered Profile

42.3.4.1 General

SMI-S conformant implementations shall provide a technique that allows clients to determine which standard the implementation conforms to. This requirement is different for RegisteredProfile instances representing an profile from SMI-S versions before 1.2.0, which are required to use the standard's version number (e.g., 1.0.3 or 1.1.0) in the RegisteredVersion property of each RegisteredProfile (or RegisteredSubprofile) instance.

Each RegisteredProfile instance representing a profile from SMI-S version 1.2.0 or later shall also be associated to a RegisteredProfile instance holding the SMI-S version number, as shown in Figure 56. The version number (RegisteredVersion) of SMI-S profiles may or may not be the same as the version number of the SMI-S Registered Profile. The RegisteredProfile instances are associated using ElementConformsToProfile where the RegisteredProfile representing SMI storage profiles (e.g., Array, Switch) is referenced from the ManagedElement role of the association. Figure 56 depicts the RegisteredProfile representing the SMI-S standard on the left, and RegisteredProfiles representing autonomous and component storage profiles in the middle.

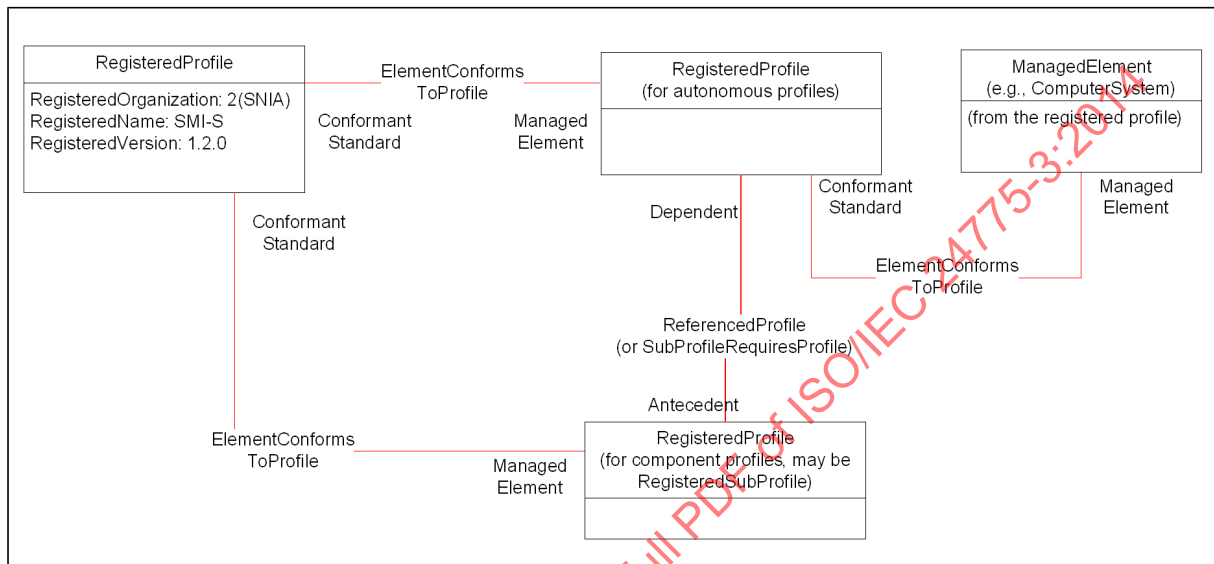


Figure 56 - Model for SMI-S Registered Profile

SMI-S class diagrams generally do not include the names of roles on associations. The requirements of roles (ConformantStandard and ManagedElement) of ElementConformsToProfile seemed critical to understand this model, so they are added to Figure 56. The role names are under the ends of the ElementConformsToProfile lines.

42.3.4.2 Provider Versions

Each RegisteredProfile and RegisteredSubprofile instance (from the Profile Registration Profile, except the SMI-S Profile) shall be associated to one (or more) SoftwareIdentity instances containing information about the software packages required to deploy the instrumentation (including providers). These are associated using ElementSoftwareIdentity. SoftwareIdentity instance may optionally be associated to Product instances representing a software product. The model for Provider Versions is depicted in Figure 57.

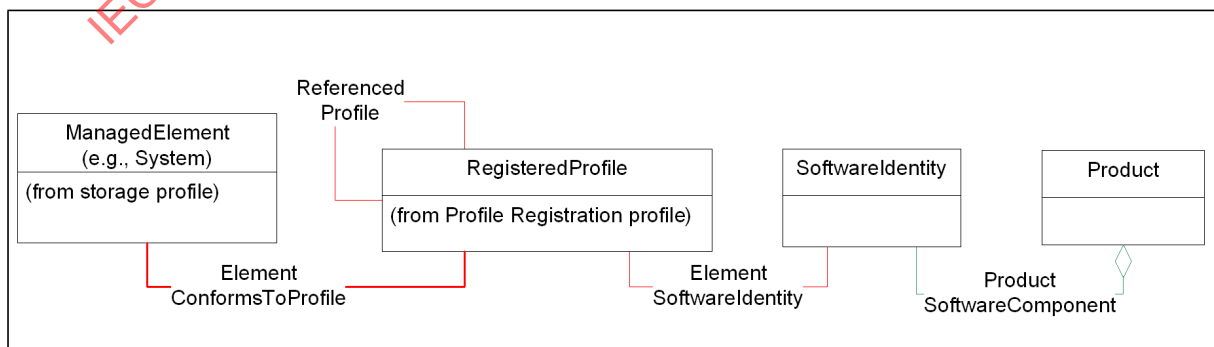


Figure 57 - Model for Provider Versions

42.3.4.3 Abstract Profile and Profile Registration

When profiles are defined for specialization, they may be defined as abstract and include this text in the Synopsis subclause:

This abstract profile specification shall not be directly implemented. Implementations shall be based on a profile specification that specializes the requirements of this profile.

RegisteredProfile instances shall not be instantiated for abstract profiles. Information about abstract profiles shall not be included in the SLP template.

42.3.4.4 Indications

The Profile Registration Profile supports optional indications for the creation and deletion of RegisteredProfile instances. These indications apply to autonomous and component profiles. The indications filters are defined in terms of the RegisteredProfile class and should be triggered for RegisteredSubprofile as well as RegisteredProfile instances. Indications should also be triggered for creation or deletion of the SMI-S profile.

These indications will be sent to subscribers when profiles are added and removed from the CIM server. They might be added due to updates, new functionality, or enabling licensed features. If an implementation supports dynamic creation or removal of profiles, then these indications should be supported.

42.3.5 Health and Fault Management Consideration

None.

42.3.6 Cascading Considerations

None

42.4 Methods

None.

42.5 Use Cases**42.5.1 Using the CIM Server Model to Determine SNIA Profiles Supported**

All SNIA profiles require the implementation of the Server Profile as part of the CIM Server. This allows a client to determine which SNIA profiles are supported by the a proxy, embedded or general purpose Information Technology – Storage Management Server. Information Technology – Storage Management clients can use SLP to search for services that support SNIA profiles. Indeed, a client may restrict its search to specific types of SNIA profiles. The client would get a response for each CIM Server service that supports a SNIA profile. From the responses, the client should use the “service-id” to determine the unique CIM Servers it is dealing with.

For each CIM Server, the client can determine the types of entities supported by inspecting the RegisteredProfilesSupported attribute returned for the SLP entries. This identifies the types of entities (e.g., devices) supported by the CIM Server.

The client may determine more detail on the support for the profiles by going to the service advertised for the CIM Server and inspecting the RegisteredProfiles maintained in the server profile. This would be done by enumerating RegisteredProfiles and RegisteredSubprofiles within the interop namespace. By inspection of the actual profile instances, the client can determine the SNIA version (RegisteredVersion) of profile, associated namespaces and associated managed elements (e.g., systems).

From the RegisteredProfiles within the namespace of the ObjectManager, a client can determine other supported profiles by following the ReferencedProfile association (or its subclass SubProfileRequiresProfile). This returns a set of RegisteredProfile (or RegisteredSubProfile) instances that represent profiles supported by the specific autonomous profile instance. See individual profile descriptions in this International Standard for the

specific list of “supported profiles”. For a given profile instance there may be zero, one or many supported profiles.

42.5.2 Recipe Assumptions

For discovery recipes, the following are assumed:

- a) a top-level object (class instance) exists for each profile, and
- b) the client knows what the top level object is.

The top-level object for each of the Information Technology – Storage Management profiles are:

- ComputerSystem: For Array, Storage (Media) Libraries, Virtualizers, Switches, and HBAs. This is the top-level ComputerSystem instance for the profile (not the component ComputerSystem or the member ComputerSystem);
- AdminDomain: For Fabric and HostDiscoveredResources;
- ObjectManager: For Server.

The top-level object (class instance) is associated to the RegisteredProfile instance for the profile via the ElementConformsToProfile association.

NOTE Other ManagedElement instances may be associated to the RegisteredProfile, but the meaning and behavior of such associations are not defined by Information Technology – Storage Management and are not mandatory.

42.5.3 Find Servers Supporting a Given Profile

```
// DESCRIPTION
// A management application wishes to find all CIM Servers on a
// particular subnet that support one or more SMI-S profiles.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1.Assume CIM Servers have advertised their services (SrvReg)
// 2.Assume there may (or may not) be Directory Agents in the subnet
// 3.Assume no security on SLP discovery
// 4.#DirectoryList[] is an array of directory URLs
// 5.#ServiceList[] is an array of service agent URLs
// 6.#DirectoryEntries [] is an array of directory entry Structures.
// The structure matches the wbem SLP Template (see Clause 5,
// section 10).

// Step 1: Set the Previous Responders List to the Null String.
#PRList = ""

// Step 2: Multicast a Service Request for a Directory Server Service.
// This is to find Directory Agents in the subnet.
//
SrvRqst (
    #PRList,          // The Previous Responders list
    "service:directory-agent" // Service type
    "DEFAULT",        // The scope
    NULL,              // The predicate
    NULL)              // SLP SPI (security token)

// Step 3: Listen for Response from Directory Agent(s)
```

```

#DirectoryList[] = DAAdvert (
    BootTimestamp, // Time of last reboot of DA
    URL,           // The URL of the DA
    ScopeList, // The scopes supported by the DA
    AttrList, // The DA Attributes
    SLP SPI List, // SLP SPI (SPIs the DA can verify)
    Authentication Block)
// Iterate on Steps 2 & 3, until a response has been received or the
// client has
// reached a UA configured CONFIG_RETRY_MAX seconds. If no DA is found,
// proceed to step 4. If a DA is found, proceed to step 7.

// Step 4: Set the Previous Responders List to the Null String
#SAPRList = ""

// Step 5: Multicast a Service Request for Service Agent Services. This
// is to find Service Agents in the subnet that are not advertised
// in a Directory.

SrvRqst (
    #SAPRList, // The Previous Responders list
    "service:service-agent" // Service type
    "DEFAULT", // The scope
    "(Service-type=WBEM)", // The predicate
    NULL) // SLP SPI (security token)

// Step 6: Listen for Response from Service Agent(s)
#SAList[] = SAAdvert (
    URL, // The URL of the SA
    ScopeList, // The scopes supported by the SA
    AttrList, // The SA Attributes
    Authentication Block)
// Iterate on Steps 5 & 6, until a response has been received or the
// client has
// reached a UA configured CONFIG_RETRY_MAX seconds. If no SA is found,
// Then record an error. There are NO WBEM SAs. Otherwise proceed to
// Step 8.

// Step 7: Unicast a Service Request to each of the DAs specifying
// a query predicate to select CIM Servers that support SNIA profiles
// and listen for responses.
for #j in #DirectoryList[]
{
    SrvRqst (
        #PRList, // The Previous Responders list
        "service:wbem", // Service type
        "DEFAULT", // The scope
        RegisteredProfilesSupported="SNIA:*" // The predicate
        NULL) // SLP SPI (security token)

    #ServiceList [#j] = SrvRply (

```

```
        Count,          // count of URLs
        URL for each SA returned)
    }
    Go to Step 9.

//Step 8: Unicast a Service Request to each of the SAs specifying
// a query predicate to select CIM Servers that support SNIA profiles
// and listen for responses.
for #j in #SAList[]
{
    SrvRqst (
        #PRList,          // The Previous Responders list
        "service:wbem",    // Service type
        "DEFAULT",        // The scope
        RegisteredProfilesSupported="SNIA:*", // The predicate
        NULL)              // SLP SPI (security token)

    #ServiceList [#j] = SrvRply (
        Count,          // count of URLs
        URL for each SA returned)
    }

// Step 9: Next retrieve the attributes of each advertisement
For #i in #ServiceList[] // for each url in list
{
    AttrRqst (
        #PRList,          // The Previous Responders list
        #ServiceList[#i], // a url from #ServiceList[]
        "DEFAULT",        // The scope
        NULL,             // Tag list. NULL means return all attributes
        NULL)             // SLP SPI (security token)
    #DirectoryEntries [#i] = AttrRply (attr-list)
    }

// Step 10: Correlate responses to the Service Request on unique
// "service-id" to determine unique CIM Servers. The client will get
// multiple responses (one for each access point) for each CIM
// Server. At this point, the client has a list of CIM Servers that
// claim to support SNIA profiles.
```

42.5.4 Enumerate Profiles Supported by a Given CIM Server

```
// DESCRIPTION
// A management application wishes to determine the Profiles supported by
// a particular CIM Server.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1.Assume the client only wants to know the "top level" profiles
// supported by the CIM Server
// 2.Assume the client has used SLP to find the CIM Servers and has a
```

```

// #DirectoryEntries [] structure
// 3.This recipe describes the operations for one of the entries in
// the #DirectoryEntries [] structure.
// 4. Assume the index into #DirectoryEntries[] for the CIM Server of
// interest is #i.

// Step 1: Get the server url for the CIM Server.
#ServerName = #DirectoryEntries[#i].service-id

// Step 2: Get the Interop Namespace for the CIM Server.
#Inamespace = #DirectoryEntries[#i].InteropSchemaNamespace[1]

// Step 3: Establish a connection to the CIM Server with
// #Inamespace. Note that the WBEM operations throughout the remainder
// of this recipe are performed with this client handle.
<Make client connection to this server using the interop namespace>

// Step 4: Get the names of all the RegisteredProfiles in the
// Interop Namespace
#ProfileName[] = EnumerateInstances("CIM_RegisteredProfile",
TRUE, TRUE, FALSE, FALSE,
["RegisteredName"])

// Step 5: Determine which RegisteredProfiles are autonomous.
// Subprofiles (aka component profiles) are associated to autonomous
// profiles via SubProfileRequiresProfile or its superclass,
// ReferencedProfile. The autonomous profile is referred to
// as the 'referencing profile' and the component/sub profile
// is referred the referenced profile. There may be more than
// two tiers, so profile may be both referenced and referencing.
// In practice, component or sub profiles would only be registered
// when their referencing autonomous profile(s) are registered, so
// any profile not referenced by another profile is autonomous.

#k = 0;
for #i in #ProfileName[i] { // walk all profiles
    $ReferencingProfiles->[] = Associators(#ProfileName[i]->,
CIM_ReferencedProfile", "CIM_RegisteredProfile", "Dependent",
"Antecedent", FALSE, FALSE, NULL);
    if ($ReferencingProfiles[] != null && $ReferencingProfiles[].length >
        0) {
        // if the profile is not referenced by another profile,
        // add it to the list of autonomous profiles
        #Autonomous[#k+1]=#ProfileName[#i]
    }
}
// #Autonomous[] now holds the autonomous RegisteredProfiles

```

42.5.5 Identify the ManagedElement Defined by a Profile

```
// DESCRIPTION
// A management application wishes to determine the ManagedElement that
// is defined by a particular Profile.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1.Assume the client has located the profile and has its object path
// ($RegisteredProfile->)

// Step 1: Determine the ManagedElement (System) by traversing the
// ElementConformsToProfile association from the RegisteredProfile
// that is the top level Profile that applies to the System
$ManagedElement->[] = AssociatorNames (
    $RegisteredProfile->,
    "CIM_ElementConformsToProfile",
    "CIM_System",
    NULL,
    NULL)

// Step 2: The object name of more than one System may be contained
// in the array returned. Examine the contents of $ManagedElement[]
// and save the name of the System of interest as $Name.

// NOTE: "Top level" object for each profile will be returned.
// To accommodate other potential ManagedElements, then it may
// be necessary need to throw out the ones that are not top level
// objects.

// NOTE: The object path for the ManagedElement may be in a Namespace
// that is different than the Interop Namespace. As a result, if the
// client wishes to actually access the ManagedElement, the client
// may get the namespace from the REF to the element:
#NameSpace=$Name.getNameSpace()
```

42.5.6 Determine the SNIA Version of a Profile

```
// DESCRIPTION
// A management application wishes to determine the SNIA version
// that a particular Profile supports.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1.Assume the client only wants to know version information
// for a SNIA profile
// 2.Assume the client has already found the profile and has the
// $RegisteredProfile-> reference

// Step 1: Get the Instance of the Profile name.
$Profile = GetInstance($RegisteredProfile->)
```

```

// Step 2: Look for an associated RegisteredProfile representing the
// SMI-S specification. This usage of RegisteredProfile was added in
// SMI-S 1.2.0, if none are found, then assume the implementation
// supports SMI-S 1.0.x or 1.1.x where the version of the profile
// matched the version of the specification. The use of ManagedElement
// and ConformantStandard as the Role and ResultRoles assure that the
// returned list is restricted to RegisteredProfiles for SMI-S spec and
// does not include domain elements.
$SpecRegisteredProfiles->[] = Associators (
    $RegisteredProfile->,
    "CIM_ElementConformsToProfile",
    "CIM_RegisteredProfile",
    ManagedElement,
    ConformantStandard,
    false,
    false,
    ["RegisteredVersion"])

if ($SpecRegisteredProfiles[] == null ||
    $SpecRegisteredProfiles[].length == 0) {
    // no RegisteredProfile for specs were found; assume the
    // version of the profile is the spec version.
    #SNIAVersion = $Profile.RegisteredVersion
} else {
    // At least one $SpecRegisteredProfile was returned; an implementation
    // may conform to multiple spec versions
    <Sort $SpecRegisteredProfile[] in reversed order of VersionNumbers>
    // The most recent supported SMI-S version is in element 0
    #SNIAVersion = $SpecRegisteredProfiles[0].RegisteredVersion
}

```

42.5.7 Find all Profiles on a Server

```

// DESCRIPTION
// A management application wishes to list all the SNIA profiles and
// their related profiles for a specific CIM Server.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1. Assume the client has already discovered the CIM Servers that
// support SNIA profiles

// Step 1: Get the names of all the RegisteredProfiles and their names
// in the Interop Namespace
$ProfileName[] = EnumerateInstances("CIM_RegisteredProfile"
    true, true, false, false, {"RegisteredName"})

// Step 2: Get the ObjectName for the Profiles
for #i in #ProfileName[] {
    $Profile->[#i]=$Name.getObjectPath(#ProfileName[#i])
}

```



```

}

// Step 3: Get the (sub)profiles associated to the profiles.
// Since ReferencedProfile is the superclass for
// SubProfileRequiresProfile and RegisteredProfile is the
// subclass for RegisteredSubProfile, this algorithm finds
// subprofiles and component profiles referenced by a
// profile.
for #i in $ProfileName[]
{
    $Subprofile[] = Associators(
        $ProfileName[#j].getObjectPath(),
        "CIM_ReferencedProfile",
        "CIM_RegisteredProfile",
        NULL, NULL, false, false, NULL)
}

```

42.6 CIM Elements

42.6.1 General

Table 402 describes the CIM elements for Profile Registration.

Table 402 - CIM Elements for Profile Registration

Element Name	Requirement	Description
42.6.2 CIM_ElementConformsToProfile (Associates Domain object (e.g. System) to RegisteredProfile)	Mandatory	Ties managed elements (e.g., Systems representing devices) to the registered profile that applies.
42.6.3 CIM_ElementConformsToProfile (Associates RegisteredProfiles for SMI-S and domain profiles)	Mandatory	Associates RegisteredProfiles for SMI-S and domain profiles.
42.6.4 CIM_ElementSoftwareIdentity (Profile and SW identity)	Mandatory	Associates a domain RegisteredProfile and SoftwareIdentity instances.
42.6.5 CIM_ElementSoftwareIdentity (Subprofile and SW identity)	Conditional	Conditional requirement: Support for instances of RegisteredSubProfile. Associates the subprofile and SoftwareIdentity instances.
42.6.6 CIM_Product	Optional	Represents a software product aggregating SoftwareIdentity instances with provider versions.
42.6.7 CIM_ProductSoftwareComponent	Optional	Associates Product and SoftwareIdentity.
42.6.8 CIM_ReferencedProfile	Optional	Associates referenced profiles using the DMTF Profile Registration profile.
42.6.9 CIM_RegisteredProfile (Domain Registered Profile)	Mandatory	An object representing a domain (e.g. Array or Switch) profile.
42.6.10 CIM_RegisteredProfile (The SMI-S Registered Profile)	Mandatory	A registered profile that provides the version of the SMI-S standard.
42.6.11 CIM_RegisteredSubProfile	Optional	Specialization of RegisteredProfile for legacy SMI-S subprofiles.
42.6.12 CIM_SoftwareIdentity	Mandatory	A representation of some bundle of providers and supporting software that shares a version number.

Table 402 - CIM Elements for Profile Registration

Element Name	Requirement	Description
42.6.13 CIM_SubProfileRequiresProfile	Optional	Specialization of ReferencedProfile referencing a SubProfile.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_RegisteredProfile	Optional	Creation of a registered profile instance. See 42.3.4.4 Indications.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_RegisteredProfile	Optional	Deletion of a registered profile instance. See 42.3.4.4 Indications.

42.6.2 CIM_ElementConformsToProfile (Associates Domain object (e.g. System) to RegisteredProfile)

The CIM_ElementConformsToProfile ties managed elements (e.g., Systems representing devices) to the registered profile that applies.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 403 describes class CIM_ElementConformsToProfile (Associates Domain object (e.g. System) to RegisteredProfile).

Table 403 - SMI Referenced Properties/Methods for CIM_ElementConformsToProfile (Associates Domain object (e.g. System) to RegisteredProfile)

Properties	Requirement	Description and Notes
ManagedElement	Mandatory	A element implementing a profile (e.g., top-level system).
ConformantStandard	Mandatory	RegisteredProfile instance describing the domain profile.

42.6.3 CIM_ElementConformsToProfile (Associates RegisteredProfiles for SMI-S and domain profiles)

Associates RegisteredProfiles for SMI-S and domain profiles.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 404 describes class CIM_ElementConformsToProfile (Associates RegisteredProfiles for SMI-S and domain profiles).

Table 404 - SMI Referenced Properties/Methods for CIM_ElementConformsToProfile (Associates RegisteredProfiles for SMI-S and domain profiles)

Properties	Requirement	Description and Notes
ManagedElement	Mandatory	The RegisteredProfile representing the domain profile.
ConformantStandard	Mandatory	The SMI-S RegisteredProfile.

42.6.4 CIM_ElementSoftwareIdentity (Profile and SW identity)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 405 describes class CIM_ElementSoftwareIdentity (Profile and SW identity).

Table 405 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity (Profile and SW identity)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to SoftwareIdentity.
Dependent	Mandatory	Reference to domain RegisteredProfile.

42.6.5 CIM_ElementSoftwareIdentity (Subprofile and SW identity)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Support for instances of RegisteredSubprofile.

Table 406 describes class CIM_ElementSoftwareIdentity (Subprofile and SW identity).

Table 406 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity (Subprofile and SW identity)

Properties	Requirement	Description and Notes
Antecedent	Mandatory	Reference to Software Identity.
Dependent	Mandatory	Reference to RegisteredSubProfile.

42.6.6 CIM_Product

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 407 describes class CIM_Product.

Table 407 - SMI Referenced Properties/Methods for CIM_Product

Properties	Requirement	Description and Notes
Name	Mandatory	Commonly used product name.
IdentifyingNumber	Mandatory	Software serial number.
Vendor	Mandatory	Product supplier.
Version	Mandatory	Product version information.

42.6.7 CIM_ProductSoftwareComponent

Associates Product and SoftwareIdentity.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 408 describes class CIM_ProductSoftwareComponent.

Table 408 - SMI Referenced Properties/Methods for CIM_ProductSoftwareComponent

Properties	Requirement	Description and Notes
GroupComponent	Mandatory	Reference to Product.
PartComponent	Mandatory	Reference to SoftwareIdentity.

42.6.8 CIM_ReferencedProfile

Associates referenced profiles using the DMTF Profile Registration profile.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 409 describes class CIM_ReferencedProfile.

Table 409 - SMI Referenced Properties/Methods for CIM_ReferencedProfile

Properties	Requirement	Description and Notes
Antecedent	Mandatory	
Dependent	Mandatory	

42.6.9 CIM_RegisteredProfile (Domain Registered Profile)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 410 describes class CIM_RegisteredProfile (Domain Registered Profile).

**Table 410 - SMI Referenced Properties/Methods for CIM_RegisteredProfile
(Domain Registered Profile)**

Properties	Requirement	Description and Notes
InstanceID	Mandatory	This is a unique value for the profile instance.
RegisteredOrganization	Mandatory	This is the official name of the organization that created the Profile. For SMI-S profiles, this would be SNIA. For DMTF profiles, this would be DMTF.
OtherRegisteredOrganization	Conditional	Conditional requirement: CIM_RegisteredProfile requires the OtherRegisteredOrganization property be populated if the RegisteredOrganization property has a value of 1 (\Other'). Mandatory if RegisteredOrganization is 1 (\Other').
RegisteredName	Mandatory	This is the name assigned by the organization that created the profile.
RegisteredVersion	Mandatory	This is the version number assigned by the organization that defined the Profile.
AdvertiseTypes	Mandatory	Defines the advertisement of this profile. If the property is null then no advertisement is defined. A value of 1 is used to indicate 'other' and a 3 is used to indicate 'SLP'.
AdvertiseTypeDescriptions	Conditional	Conditional requirement: CIM_RegisteredProfile requires the AdvertiseTypeDescriptions property be populated if the AdvertiseTypes property has a value of 1 (\Other'). This shall not be NULL if 1 (\Other') is identified in AdvertiseType.

42.6.10 CIM_RegisteredProfile (The SMI-S Registered Profile)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 411 describes class CIM_RegisteredProfile (The SMI-S Registered Profile).

**Table 411 - SMI Referenced Properties/Methods for CIM_RegisteredProfile
(The SMI-S Registered Profile)**

Properties	Requirement	Description and Notes
InstanceID	Mandatory	A unique value for the profile instance.
RegisteredOrganization	Mandatory	Shall be 11 (SNIA).
RegisteredName	Mandatory	Shall be 'SMI-S'.

**Table 411 - SMI Referenced Properties/Methods for CIM_RegisteredProfile
(The SMI-S Registered Profile)**

Properties	Requirement	Description and Notes
RegisteredVersion	Mandatory	The version number of the SMI specification the associated profiles conform to.
AdvertiseTypes	Mandatory	Should be 2 (Not Advertised) or 3 (SLP). 2 is recommended to avoid increasing size of SLP template.
AdvertiseTypeDescriptions	Conditional	Conditional requirement: CIM_RegisteredProfile requires the AdvertiseTypeDescriptions property be populated if the AdvertiseTypes property has a value of 1 (\Other\). This shall not be NULL if 'Other' is identified in AdvertiseType.

42.6.11 CIM_RegisteredSubProfile

Specialization of RegisteredProfile for legacy SMI-S subprofiles.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 412 describes class CIM_RegisteredSubProfile.

Table 412 - SMI Referenced Properties/Methods for CIM_RegisteredSubProfile

Properties	Requirement	Description and Notes
InstanceID	Mandatory	This is a unique value for the subprofile instance.
RegisteredOrganization	Mandatory	This is the official name of the organization that created the subprofile. For SMI-S profiles, this would be 11 ('SNIA').
OtherRegisteredOrganization	Conditional	Conditional requirement: CIM_RegisteredProfile requires the OtherRegisteredOrganization property be populated if the RegisteredOrganization property has a value of 1 (\Other\). Mandatory if RegisteredOrganization is 1 ('Other').
RegisteredName	Mandatory	This is the name assigned by the organization that created the subprofile.
RegisteredVersion	Mandatory	This is the version number assigned by the organization that defined the subprofile.
AdvertiseTypes	Mandatory	Should be 2 (Not Advertised) for subprofiles.
AdvertiseTypeDescriptions	Conditional	Conditional requirement: CIM_RegisteredProfile requires the AdvertiseTypeDescriptions property be populated if the AdvertiseTypes property has a value of 1 (\Other\). This shall not be NULL if 1 ('Other') is identified in AdvertiseType.

42.6.12 CIM_SoftwareIdentity

Created By: Static

Modified By: Static
Deleted By: Static
Requirement: Mandatory.

Table 413 describes class CIM_SoftwareIdentity.

Table 413 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity

Properties	Requirement	Description and Notes
Name	Mandatory	A user-friendly name for the instrumentation software.
InstanceID	Mandatory	
VersionString	Mandatory	
Manufacturer	Mandatory	The name of the company associated with the instrumentation software.
Classifications	Mandatory	
ClassificationDescriptions	Conditional	Conditional requirement. CIM_SoftwareIdentity requires the ClassificationDescriptions property be populated if the Classifications property has a value of 1 ('Other'). Mandatory if Classifications is set to 1 ('Other').

42.6.13 CIM_SubProfileRequiresProfile

Specialization of ReferencedProfile referencing a SubProfile.

Created By: Static
Modified By: Static
Deleted By: Static
Requirement: Optional.

Table 414 describes class CIM_SubProfileRequiresProfile.

Table 414 - SMI Referenced Properties/Methods for CIM_SubProfileRequiresProfile

Properties	Requirement	Description and Notes
Dependent	Mandatory	Reference to a RegisteredSubProfile.
Antecedent	Mandatory	Reference to a RegisteredProfile.

STABLE

STABLE**43 Indication Profile****43.1 Description****43.1.1 General**

The Indication Profile is a component profile of the Server Profile. It may also be a component profile of any other profile (e.g., Array Profile).

Refer to individual profile definitions to see whether or not the Indication Profile is mandatory. Figure 58 illustrates the structure of profiles, the Indication Profile and indication instances implied by an Array's support for the Indication Profile.

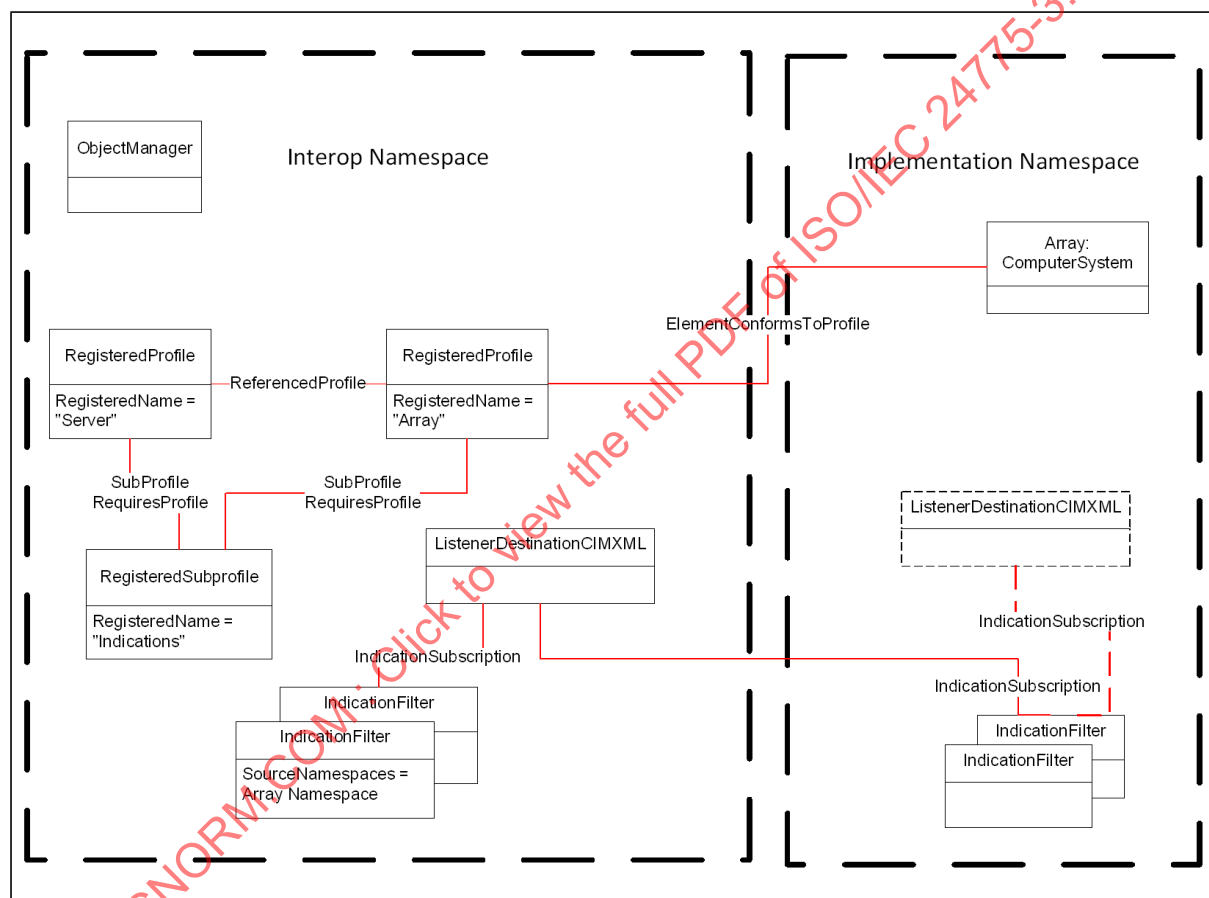


Figure 58 - Indication Profile and Namespaces

Indication filters are defined in the context of the namespace in which they are implemented. In Figure 58, this is shown as the implementation namespace. The indication filters shall be defined in two places: The Interop namespace and the namespace where the indications are intended to originate.

DEPRECATED

For the Filters defined in the InteropNamespace, the SourceNamespaces property shall be filled out to indicate the implementation namespace where the indications are to originate. For the

IndicationFilters defined in the implementation namespace, this property may be null (indicating the indications originate in the implementation namespace of the array).

DEPRECATED

EXPERIMENTAL

The SourceNamespaces property of an IndicationFilter indicates the list of namespaces from which indications for the Filter query are to originate (e.g., the implementation namespace for the device). For any filter, the SourceNamespaces property should be populated. For the Filters in the InteropNamespace, the Namespaces properties would indicate the implementation namespaces that support the indications defined by the filter. For filters defined in the implementation Namespace, the filters should contain the Implementation Namespace (as one of its source namespaces).

EXPERIMENTAL

The IndicationFilters may be populated by the Provider (or they may be created by a client). In either case, they are created in both the Interop Namespace and the implementation namespace of the array. The ListenerDestinationCIMXML class shall be in the Interop Namespace and may also be in the implementation namespace. And there would be two instantiations of the IndicationSubscription association: one in the Interop Namespace and one in the implementation namespace.

SMI-S profile implementations that support indications shall support either the use of "predefined" indications filters, "client defined" indication filters or both. In the case of an implementation that supports "predefined" filters, the SMI-S Server would populate its model with indication filters that it supports. SMI-S Clients would select the indication filters to which they wish to subscribe from the list supplied by the SMI-S Server (enumeration of IndicationFilters in the appropriate namespace). In the case of an implementation that supports "client defined" filters, the SMI-S Server shall support filter creation (and deletion) by clients and it shall support creation of at least the filters defined by the profile.

Creation of an IndicationFilter will cause the creation of instances in both the InteropNamespace and the implementation namespace. ListenerDestinationCIMXML instances should be created in the InteropNamespace, but may also be created in the implementation namespace (for SMI-S 1.0.x compatibility reasons). If a ListenerDestinationCIMXML instance is created in the implementation namespace, a duplicate instance will be instantiated in the InteropNamespace. However, if a ListenerDestinationCIMXML is created in the InteropNamespace, it may not be created in the implementation namespace.

NOTE An implementation may support both "predefined" filters and "Client Defined" filters.

SMI-S Clients would subscribe to the indications for the events to which they wish to be notified. They would also supply an address (Indication listener) in which the indications are to be sent. SMI-S Clients shall use the subclass ListenerDestinationCIMXML when creating subscriptions.

In any given implementation Indication Filters are scoped by NameSpace. That is, a subscription to the change of operational status for a ComputerSystem can result in reporting of any change of operational status for ANY ComputerSystem managed within a Namespace. A client should inspect any indication to see if it is for an element that it manages.

A vendor implementation may support additional indication filters beyond those identified in a profile specification, but all the filters identified in SMI-S shall be supported as specified by the profile.

Note that indication filters may correspond to optional or conditional features in a profile. When a provider supports an optional or conditional feature, the indications corresponding to the feature may be conditional on the feature. This means that the provider shall supply the filters or shall allow a client to define the filters. But optional indications that correspond to the feature need not be supported. Indications corresponding to the filter shall be generated by the provider when a corresponding event occurs. On the other hand, if a profile implementation does not support a component profile that defines mandatory indications, then the profile implementation does not need to support those indications.

EXPERIMENTAL

43.1.2 IndicationFilter Names

IndicationFilters have a Name property. The value of the Name property in instances defined by referencing profiles should be formatted as defined by the following ABNF rule:

OrgID ":" RegisteredName ":" UniqueID

Where OrgID identify the business entity owning the referencing profile. OrgID shall include a copyrighted, trademarked, or otherwise unique name that is owned by that business entity or that is a registered ID assigned to that business entity by a recognized global authority. In addition, to ensure uniqueness, OrgID shall not contain a colon (:).

For referencing profiles owned by the SNIA, OrgID should match "SNIA" for IndicationFilters defined by the International Standard. For client defined IndicationFilters, the OrgID should identify the client (application) organization.

The RegisteredName should be the registered name of the referencing profile, as defined by the value of its CIM_RegisteredProfile.RegisteredName property.

The UniqueID should uniquely identify the instance within the referencing profile.

EXPERIMENTAL

43.1.3 Basic Indication Classes and Association

Figure 59 illustrates the classes used in support of indications. Any given profile implementation may not include all of these classes. But they would at least support IndicationFilters (possibly predefined), ListenerDestinationsCIMXML and IndicationSubscriptions. The actual types of indications supported can vary by profile (see 43.8 "CIM Elements" to determine the types of indications supported).

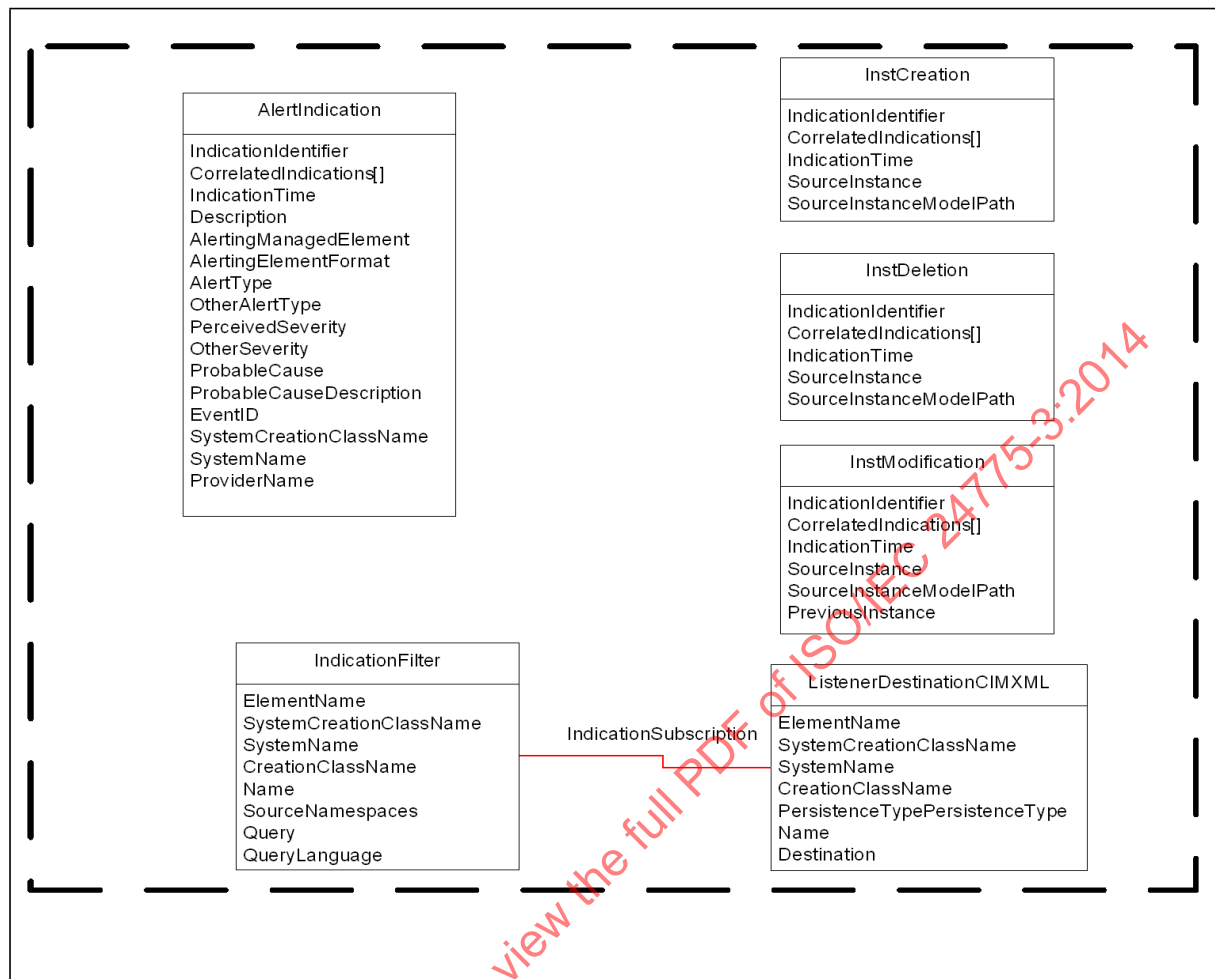


Figure 59 - Indication Profile Instance Diagram

Clients request indications to be sent to them by subscribing to the indication filters. Subscriptions are stored in the SMI-S Server. A Subscription is expressed by the creation of an IndicationSubscription association instance that references an IndicationFilter (a filter) instance, and an ListenerDestination (for the handler of the indications) instance. A Filter contains the query that selects an indication class or classes.

SMI-S Servers that support SMI-S profiles that provide CIM indications support shall populate their models with the filters as defined by the profile(s) or allow clients to create the filters that are defined for the profile(s). Additional filters may also be created by indication consumers (e.g., SMI-S Clients), but this is not mandatory with SMI-S. The client would create these filters using CreateInstance intrinsic method.

The query property of the IndicationFilter is a string that specifies which indications are to be delivered to the client. There is also a query language property that defines the language of the query string. Example query strings are:

“SELECT * FROM AlertIndication”

“SELECT * FROM InstModification WHERE SourceInstance ISA ComputerSystem”

AlertIndication and InstModification are types of indications. The first query says to deliver all alert type indications to the client, and the second query says to deliver all instance modification indications to the client, where the instance being modified is a ComputerSystem (or any subclass thereof).

See Annex C (normative) Indication Filter Strings in *Information Technology – Storage Management, Part 2 Common Architecture* for information on the use of indication filter strings.

A ListenerDestination specifies the means of delivering indications to the client. The subclass ListenerDestinationCIMXML provides for XML encoded indications to be sent to a specific URL, which is specified as a property of that class.

When a client receives an indication, it will receive some information with the indication, and then it may need to do additional queries to determine all of the consequences of the event.

Note that to avoid multiple calls to get additional data for an indication, profile designers (or clients, for client defined filters) should consider more elaborate Queries for Filters to return more information.

The instances of AlertIndications, InstCreation, InstDeletion and InstModification are temporary. They exist until they are delivered to the subscribing clients. The ListenerDestinationCIMXML, IndicationFilter and IndicationSubscription instance are permanent. That is, they persist until action is taken by client to delete them.

One final note on the indications supported. InstModification may or may not require the PreviousInstance property. A profile may be designed to require it or not. If the SMI-S profile defines an IndicationFilter on InstModification it shall specify whether or not PreviousInstance is required. It may always be recommended. If a profile defines PreviousInstance as optional, then an implementation may provide a previous instance (or not). However, if the SMI-S profile defines an IndicationFilter on InstModification with PreviousInstance required, then all implementations shall implement the PreviousInstance property.

43.1.4 Life Cycle Indications

A life cycle indication is used to convey changes in the model. It is represented by a subclass of InstIndication. Life cycle indications are concerned only with the creation, modification, or deletion of CIM Instances. The indication is a CIM class whose properties contain copies of CIM Instances that have been created, modified, or deleted (InstCreation, InstDeletion, InstModification). As such, life cycle indications can only report on classes that are supported. Profile designers use life cycle indications as means where clients can monitor for significant changes in a particular data model. The significant changes to the model are a reflection of changes in the managed element the CIM instance(s) represents. An event like component overheat is likely to affect several components. Therefore, in many cases the scope of the event can be observed through the telemetry communicated through life cycle indications.

The mandatory properties of a life cycle indication are:

- IndicationTime - The time and date of creation of the Indication.
- SourceInstance - A copy of the instance that changed to generate the Indication. SourceInstance contains the current values of the properties selected by the Indication Filter's Query.
- SourceInstanceModelPath - The Model Path of the SourceInstance.

In addition, the following properties are recommended, but not mandatory:

- IndicationIdentifier - An identifier for the Indication that may be used for correlated indications.
- CorrelatedIndications - IndicationIdentifiers whose notifications are correlated with this one.

In addition, for InstModification indications, the PreviousInstance property may be provided:

- PreviousInstance - A copy of the 'previous' instance whose change generated the Indication. PreviousInstance contains 'older' values of an instance's properties (as compared to SourceInstance), selected by the IndicationFilter's Query.

43.1.5 AlertIndications

An alert indication is another type of indication but with a different purpose. Alert Indications are used to draw attention of subscribing client applications to the occurrence of an event. Alert Indication may describe aspects of an event. The event's characteristics may or may not be wholly or partially represented in the data model (as expressed through CIM). An Alert Indication is represented by a subclass of AlertIndication. The alert indication itself is considered a change in the instrumentation's model.

An alert indication describes the category, severity, and event specifics. However, the event specifics may not be understandable by an SMI-S Client. A standard message can convey the event specifics in a manner defined by SMI-S or another related standard. (See *Information Technology – Storage Management, Part 2 Common Architecture 9 Standard Messages*.) The interpretation for the alert indication is either contained within a standard message registry that is referenced by a profile, or defined by the profile to be produced for some reason and identifiable in some manner.

The mandatory properties of an AlertIndication are:

- IndicationTime - The time and date of creation of the Indication.
- AlertingManagedElement - The identifying information of the entity (i.e., the instance) for which this Indication is generated. The property contains the path of an instance, encoded as a string parameter - if the instance is modeled in the CIM Schema. If not a CIM instance, the property contains some identifying string that names the entity for which the Alert is generated.
- AlertingElementFormat - The format of the AlertingManagedElement property is interpretable based upon the value of this property. Values are defined as: "Unknown", "Other", "CIMObjectPath"
- AlertType - This is an integer property that is a value map. The values supported are: "Other", "Communications Alert", "Quality of Service Alert", "Processing Error", "Device Alert", "Environmental Alert", "Model Change", "Security Alert"
- PerceivedSeverity - An enumerated value that describes the severity of the Alert Indication from the notifier's point of view. This is an integer property that is a value map. The values supported are: "Unknown", "Other", "Information", "Degraded/Warning", "Minor", "Major", "Critical", "Fatal/Non Recoverable".
- ProbableCause - This is an integer property that is a value map. There are many values that may be set (refer to the MOF for details).
- SystemCreationClassName - The scoping System's CreationClassName for the Provider generating this Indication.
- SystemName - The scoping System's Name for the Provider generating this Indication.
The SystemName would typically be the same that for a system in the Implementation Namespace (unless the Indication is an indication generated for Server Profile).
- ProviderName - The name of the Provider generating this Indication.

In addition, the following properties are recommended, but not mandatory:

- IndicationIdentifier - An identifier for the Indication that can be used for identification when correlating Indications (see 43.1.7.2 "Indication Identification").
- CorrelatedIndications[] - IndicationIdentifiers whose notifications are correlated with this one.
- Description - A short description of the Indication.
- OtherAlertType - This property is mandatory if the AlertType is 1 (for "other").
- OtherSeverity - This property is mandatory if the PerceivedSeverity is 1 (for "other")

- ProbableCauseDescription - Provides additional information related to the ProbableCause.
- EventID - An instrumentation or provider specific value that describes the underlying "real-world" event represented by the Indication.
- OwningEntity - A string that uniquely identifies the entity that owns the definition of the format of the Message. For messages owned by the SNIA, this shall be encoded as 'SNIA'. However, for SMI-S, not all messages need be owned by SNIA.
- MessageID - A string that uniquely identifies, within the scope of the OwningEntity, the format of the Message.
- Message - The formatted message (including the MessageArguments).
- MessageArguments - An array of strings that contain the dynamic content of the message.

For descriptions on how these properties should be encoded, see the profile for specific alert indications that are supported. For encoding of the OwningEntity, MessageID, Message, and MessageArguments of SNIA messages, see 9.3 in the *Information Technology – Storage Management, Part 2 Common Architecture*.

43.1.6 Indication Delivery

Acceptance of a subscription, represented by an instance of the IndicationSubscription association between an instance of IndicationFilter and an instance of ListenerDestination, is a contract between the SMI-S Server and SMI-S Client that requires that the SMI-S Server shall produce indications when the conditions described by the associated indication filter are present. The SMI-S Server may not be able to deliver the indication for other reasons like authentication failures or network connectivity failures, but the SMI-S Server shall attempt to deliver the indication.

In some cases, the Client (ListenerDestination) may not be available when an event occurs that requires delivery to the client. In such cases, the CIM Server should attempt delivery to the listener destination 3 times. If the delivery cannot be made within 3 attempts, the indication may be considered delivered.

If the ListenerDestinationCIMXML.PersistenceType is set to "3" (transient), the IndicationSubscription may be deleted after 3 attempts that fail. If the ListenerDestinationCIMXML.PersistenceType is set to "2" (permanent) the IndicationSubscription shall be retained.

43.1.7 Instrumentation Requirements

43.1.7.1 General Instrumentation Considerations

A SMI-S Server may allow a client to create indications filters. If the SMI-S Server does not support this option, then the server shall send a return code indicating a request to create an instance of a filter is unsupported. This allows the provider to inform clients which types of indications the provider supports. For example, a provider that does not support SNMPTrapAlertIndications shall return unsupported for an indications filter create request.

43.1.7.2 Indication Identification

Not defined in this version of the Indication Profile.

43.1.7.3 SMI-S Dedicated Server Considerations

The dedicated server should supply more detailed queries as described in the profile sections.

A standard implementation of indications requires the server to accept client requests to create ListenerDestinations. The dedicated server implementation uses the Instance Manipulation functional group in addition to Basic Read.

43.1.7.4 Additional Indications

Most Indication Filters defined in the “CIM Elements” section of the International Standard are mandatory. However, a profile may also document additional Indication Filters as optional filters. A client can determine whether or not “additional” indication filters are supported by one of two techniques:

- 1) Enumerating Predefined Indication Filters – this will return all the indication filters that have been predefined by the provider for the Namespace.
- 2) CreateInstance of the desired “additional” Indication Filter – if the “additional” indication filter is supported, the CreateInstance will succeed.

43.1.7.5 Support for Query Languages

For versions of the standard prior to 1.3.0, CQL had not been approved as a standard and was treated as recommended and experimental. For those early versions, WQL (also referred to as the SMI-S query Language) was the non-experimental query language.

For versions of the standard starting at 1.3.0, CQL is mandatory for newly defined indication filters. WQL alternatives shall not be defined in the standard.

DEPRECATED

Support for the SMI-S 1.0.x Query Language is being deprecated.

DEPRECATED

See Annex C (normative) Indication Filter Strings in *Information Technology – Storage Management, Part 2 Common Architecture* for information on the use of indications filter strings.

43.1.7.6 Timing of Delivery of Indications

There are no standards for how quickly an implementation shall deliver an indication. All reasonable attempts should be made by the implementation to deliver all indications at the CIM Server’s earliest convenience.

There are also no standard guidelines on how long or how many attempts should be made to deliver an indication. As a general guideline an implementation should make at least 3 attempts to deliver an indication before giving up trying to deliver the indication. Similarly, delivery of indications should allow at least 30 s to elapse before giving up trying to deliver the indication. The intent is to allow sufficient time to allow any network problems to clear.

43.1.7.7 Handling of Indication Storms

43.1.7.7.1 General

Occasionally an event may occur that causes many indication filters to evaluate to true (an trigger many indications). This situation is referred to as an “indication storm.” These can be very expensive and degrade the performance of the environment. To contain the impact of this implementation can employ the following technique:

- use the RepeatNotificationPolicy (and related properties) of the IndicationSubscription.

43.1.7.7.2 Use of RepeatNotificationPolicy

The RepeatNotificationPolicy property defines the desired behavior for handling Indications that report the occurrence of the same underlying event (e.g., the disk is still generating I/O errors and has not yet been repaired). For SMI-S, this is extended to include multiple indications that are generated from a single IndicationFilter.

The related properties are RepeatNotificationCount, RepeatNotificationInterval, and RepeatNotificationGap. The defined semantics for these properties depend on the value of

RepeatNotificationPolicy, but values for these properties shall be set if the property is defined for the selected policy.

If the value of RepeatNotificationPolicy is 2 (None), then the client will receive all indications. Special processing of repeat Indications shall not be performed.

If the value is 3 (Suppress) the first RepeatNotificationCount Indications, describing the same event, shall be sent and all subsequent Indications for this event suppressed for the remainder of the time interval RepeatNotificationInterval. A new interval starts when the next Indication for this event is received. That is, all indications after the first 'n' (where 'n' is defined by the RepeatNotificationCount) are not sent (within the RepeatNotificationInterval time).

If the value of RepeatNotificationPolicy is 4 (Delay), then indications are collected and notification is only sent after a certain number of events happen (as defined by RepeatNotificationCount) or the time interval (RepeatNotificationInterval) lapses. When an event happens, the Indication shall be suppressed if, including this Indication, RepeatNotificationCount or fewer Indications for this event have been received during the prior time interval defined by RepeatNotificationInterval. If this Indication is the RepeatNotificationCount + 1 Indication, this Indication shall be sent and all subsequent Indications for this event ignored until the RepeatNotificationGap has elapsed. A RepeatNotificationInterval may not overlap a RepeatNotificationGap time interval.

For SMI-S, a single indication filter that identifies a change in OperationalStatus on StorageVolumes would be subjected to the RepeatNotificationPolicy, even though the repeat notifications may be from multiple StorageVolumes.

The RepeatNotificationPolicy can vary by implementation (or even IndicationFilter). However, it shall be specified on any subscription. The valid values for an SMI-S implementation are:

- 2 (None),
- 3 (Suppress), or
- 4 (Delay).

An SMI-S profile may restrict this further for any given indication filter, but it cannot expand this to other policies without breaking interoperability. For example, a profile might restrict InstCreation filters for ComputerSystems to "None" and restrict InstModification filters on StorageVolume to "Suppress" or "Delay." But an SMI-S profile shall not define "unknown" as a valid SMI-S setting for the RepeatNotificationPolicy.

NOTE RepeatNotificationPolicy set to 2 "none" is compatible with SMI-S 1.0.

43.1.7.8 Clarification of indication generation

43.1.7.8.1 General Requirements

To minimize the use of stale object references by WBEM Clients, a WBEM Server shall generate instance deletion indications, where defined as mandatory profile elements, whenever a MSE instance is removed while the WBEM Server is operational. The indication shall be generated for all causes of removal, which include but are not limited to, explicit WBEM instance manipulation by some WBEM Client, internal implementation of the WBEM Server outside the scope of SMI-S, and a side effect of invoking some WBEM extrinsic method.

A WBEM Server should generate instance deletion indications, where defined as mandatory profile elements, whenever a MSE instance that was present before a failure of the device or application is no longer present when the device or application recovers from the failure.

NOTE SMI-S already requires WBEM Servers to persist WBEM Client subscription for indications.

A WBEM Server shall generate instance creation indications, where defined as mandatory profile elements, whenever a MSE instance is created while the WBEM Server is operational. A WBEM Server shall also generate instance creation indications, where defined as mandatory

profile elements, whenever a MSE instance that was not present before a failure of the device or application is present when the device or application recovers from the failure.

Almost universally in SMI-S profiles, all MSEs can be linked by association back to a specific "top-level" MSE. In most profiles this is either a ComputerSystem or a AdminDomain. A WBEM Server that is providing information on multiple devices will have multiple MSE instances, one for each of the devices. The behavior of WBEM Operations in the face of a failure of the device or applications differs.

43.1.7.8.2 Definition of "failed" MSE

A MSE instance is defined to be failed if any of the following conditions hold:

- 1) Failure status are contained in the OperationalStatus attribute, when present, and OperationalStatus array does not contain "OK"
- 2) EnumerateInstances, EnumerateInstanceNames, Associators, AssociatorNames, References, ReferenceNames WBEM Operations might return meaningless or no information for any mandatory profile element. OperationalStatus when present in the class will have meaningful data and will have a failure status. Explicit values for "unknown" or "undetermined" are completely meaningful when defined for a profile element.
- 3) WBEM extrinsic operations that ERR_FAILED may indicate that this instance is failed.
- 4) CIM Instances that were returned before the failure of the MSE might not be returned after the failure. Indications representing the OperationalStatus change to a failure status were produced for the this 'top-level' CIM Instance or 'top-level' parent CIM Instance. The combination of these two situations define failure in this case.

A MSE with an OperationalStatus of "Lost Communications" or "No Contact" obviously shall be considered failed because no WBEM operations can succeed.

An OperationalStatus of "Starting", "Stopping", or "Stopped" does not mandate failure. The detailed behavior of the MSE with regard to the conditions given above, determines whether these status's indicate failure. The WBEM Client should be warned of a possible failure scenario when receiving these status.

43.1.7.8.3 Minimal function for failed MSEs

Any failed instance represented by any WBEM Server shall support the following functionality. If the WBEM Server is not able to support the functionality on a failed instance, it shall delete the instance.

- 1) EnumerateInstances, EnumerateInstanceNames, Associators, AssociatorNames, References, and ReferenceNames WBEM Operations that include the failed instance as part of the return set will complete without error. The Key and the OperationalStatus attributes, when present, shall be properly provided.
- 2) When a GetInstance WBEM Operation is attempted on the failed instance, CIM_ERR_FAILED shall be returned with a message describing or indicating the failure of the device or application.
- 3) Failed instance names shall be returned from WBEM Operations that return Object Names. Failed instances shall be returned for WBEM Operations that return Instances but only the keys and OperationalStatus, when present, are mandatory.
- 4) Method invocations on failed MSEs will fail with the CIM_ERR_FAILED error.

43.1.7.8.4 Isolation of failed top-level MSEs

For efficiency and consistency of navigation, a WBEM Client should not be able to retrieve false or meaningless information from the WBEM Server about a MSE instance.

A WBEM Server can take one of two actions in the Failed MSE case and top-level MSE instances. It shall set the OperationalStatus on the top-level MSE instance to reflect the failed

state and forward the related CIM Indications as required. It may also remove all directly or indirectly associated instances, generating the corresponding indications.

A WBEM Client shall be prepared to deal with a WBEM Object CIM_ERR_NOT_FOUND error, indicating the use of a stale object reference not avoided by timely receipt and processing of an instance deletion indication. A WBEM Client shall also consider the OperationalStatus of any MSE for which OperationalStatus is a mandatory profile element before treating the other attributes and associations of the instance as meaningful.

43.1.7.9 HTTP Security

Not defined in this version of the International Standard.

43.2 Health and Fault Management Considerations

43.2.1 Elements Reporting Health

The Indication Profile has no classes that report health information. However, indications are a means available for reporting changes in health status.

43.2.2 Health State Transformations and Dependencies

No Indications class have OperationalStatus or HealthState properties.

43.2.3 Standard Errors Produced

All manipulation of Indication classes and associations are done using intrinsic methods. The errors produced are those listed for intrinsic methods.

43.2.4 Cause and effect associations

Cause and effect associations are defined as part of the Health and Fault Management Package.

EXPERIMENTAL

43.2.5 Indication Correlation

Not defined in this version of the International Standard.

EXPERIMENTAL

43.3 Cascading Considerations

Not Applicable.

43.4 Supported Profiles, Subprofiles and Packages

Related Profiles for Indication: Not defined in this International Standard.

43.5 Methods of the Profile

43.5.1 Extrinsic Methods of the Profile

No extrinsics are specified on the Indication Profile.

43.5.2 Intrinsic Methods of the Profile

The Indication Profile is mostly populated by providers and is accessible to clients using basic read and association traversal. However, there are two constructs that would be created by Clients. These are the ListenerDestinationCIMXML and the IndicationSubscription. In addition, a client may be able to create an IndicationFilter. In addition to being able to create them, client may delete them (except "pre-defined" filters which cannot be deleted), and a client may

modify any IndicationFilter that was client created. These functions are performed using the intrinsics:

Table 415 shows Indication Profile methods that cause Instance Creation, Deletion or Modification.

Table 415 - Indication Profile Methods that cause Instance Creation, Deletion or Modification

Method	CreatedInstances	Deleted Instances	Modified Instances
CreateInstance	ListenerDestinationCIMXML	N/A	N/A
CreateInstance	IndicationSubscription	N/A	N/A
CreateInstance	IndicationFilter	N/A	N/A
DeleteInstance	N/A	ListenerDestinationCIMXML	N/A
DeleteInstance	N/A	IndicationSubscription	N/A
DeleteInstance	N/A	IndicationFilter	N/A
ModifyInstance	N/A	N/A	IndicationFilter

CreateInstance - for ListenerDestinationCIMXML, IndicationSubscription and IndicationFilter

```
<instanceName>CreateInstance (
    [IN] <instance> NewInstance
)
```

EXPERIMENTAL

An implementation should populate all fields (scheme, hostname, port number, namespace, key) in object path for the instance being created.

The host name portion shall be set to a valid, client-resolvable host name (i.e., DNS) or IPv4 or IPv6 address. Internal names (e.g., /etc/hosts) are not valid. If host name, then must be FQDN (not a short name). This implies the provider shall be configured as a valid DNS client to use host names and cannot rely on an administratively defined name.

If an implementation supports CreateInstance on IndicationFilters or ListenerDestinations, the implementation should allow NULL to be specified for key properties. If key properties are passed in on the CreateInstance, the implementation may ignore the keys.

For IndicationFilters and ListenerDestinations, if the client supplies an ElementName, the implementation shall persist the property for later use by the client.

For IndicationFilters the Query strings are case sensitive and the implementation shall preserve case.

EXPERIMENTAL

If successful, the return value defines the object path of the new CIM Instance relative to the target Namespace (i.e., the Model Path), created by the CIM Server.

Note that for CreateInstance of an IndicationSubscription requires that the ListenerDestinationCIMXML instance and the IndicationFilter exist.

If unsuccessful, one of the following status codes shall be returned by this method, where the first applicable error in the list (starting with the first element of the list, and working down) is

the error returned. Any additional method-specific interpretation of the error in is given in parentheses.

CIM_ERR_ACCESS_DENIED, CIM_ERR_NOT_SUPPORTED, CIM_ERR_INVALID_NAMESPACE, CIM_ERR_INVALID_PARAMETER (including missing, duplicate, unrecognized or otherwise incorrect parameters), CIM_ERR_INVALID_CLASS (the CIM Class of which this is to be a new Instance does not exist), CIM_ERR_ALREADY_EXISTS (the CIM Instance already exists), CIM_ERR_FAILED (some other unspecified error occurred).

Note that a ListenerDestinationCIMXML instance should be created in the Interop namespace. However, they may be created in a "Source" namespace. If the client creates a ListenerDestinationCIMXML instance in a "Source" namespace, then a duplicate ListenerDestinationCIMXML instance shall be created in the Interop Namespace, if it does not already exist in the Interop Namespace.

NOTE 1 The inverse is not true. If the client creates the ListenerDestinationCIMXML instance in the Interop Namespace, no instance will be created in other namespaces (there is nothing that would indicate which namespaces would be the Source namespaces).

IndicationFilters may be created in either the Interop Namespace or an implementation namespace in which the indications are to originate. In either case, the Client only needs to create one instance (and providers will automatically create the corresponding instance in the other namespaces).

NOTE 2 If a client attempts to create an IndicationFilter that already exists (has the same key fields), but other properties are different, then the request will fail. If the Client attempts to create an IndicationFilter that has identical properties to an existing IndicationFilter instance, it will succeed and CreateInstance need not treat the instance as a separate instance.

When a client creates an IndicationSubscription the client only needs to create a subscription to one of the IndicationFilters (the provider will automatically generate the corresponding subscription to the other filter instance). Even though there are two instance of the IndicationFilter created (and two instances of the subscription) duplicate indications will not be sent to the ListenerDestination.

Indeed, in general, redundant subscriptions need not produce duplicate indications (that is, if the same listener subscribes to two filters that are equivalent, then an implementation need not produce two indications).

DeleteInstance - for ListenerDestinationCIMXML, IndicationSubscription and IndicationFilter

```
void DeleteInstance (
    [IN] <instanceName> InstanceName
)
```

The InstanceName input parameter defines the name (model path) of the Instance to be deleted.

If successful, the specified Instance (ListenerDestinationCIMXML, IndicationSubscription or IndicationFilter) shall have been removed by the CIM Server.

The deletion of a ListenerDestinationCIMXML or an IndicationFilter instance will cause the automatic deletion of any associated IndicationSubscription instances. Deletion of an IndicationSubscription will not cause the deletion of any corresponding ListenerDestinationCIMXML or IndicationFilter instances. For example, the deletion of an instance may cause the automatic deletion of all associations that reference that instance. Or the deletion of an instance may cause the automatic deletion of instances (and their associations) that have a Min(1) relationship to that instance.

If unsuccessful, one of the following status codes shall be returned by this method, where the first applicable error in the list (starting with the first element of the list, and working down) is the error returned. Any additional method-specific interpretation of the error in is given in parentheses.

CIM_ERR_ACCESS_DENIED, CIM_ERR_NOT_SUPPORTED, CIM_ERR_INVALID_NAMESPACE, CIM_ERR_INVALID_PARAMETER (including missing, duplicate, unrecognized or otherwise incorrect parameters), CIM_ERR_INVALID_CLASS (the CIM Class does not exist in the specified namespace), CIM_ERR_NOT_FOUND (the CIM Class does exist, but the requested CIM Instance does not exist in the specified namespace), CIM_ERR_FAILED (some other unspecified error occurred).

DEPRECATED

NOTE 3 Deleting the instance of an IndicationFilter in the Interop Namespace will cause the corresponding IndicationFilter in the "SourceNamespace" to also be deleted (and vice versa). Deletion of an indication filter will also cause all subscriptions to that filter to be deleted. However, deletion of a filter will not cause the deletion of any listener destination.

NOTE 4 Deleting the instance of an IndicationSubscription in the InteropNamespace will cause the corresponding IndicationSubscription in the "SourceNamespace" to also be deleted (and vice versa). However, deleting a subscription will not delete filters or listener destinations.

NOTE 5 Deleting the instance of ListenerDestinationCIMXML in either the InteropNamespace or the "source" namespace will cause the corresponding instance (if one exists) to be deleted.

DEPRECATED

EXPERIMENTAL

NOTE 6 Deleting the instance of an IndicationFilter in the Interop Namespace will cause the corresponding IndicationFilters in the "SourceNamespaces" to also be deleted. Deletion of an IndicationFilter in one of the SourceNamespaces will not cause the deletion of the IndicationFilter in the InteropNamespace, unless it is the "last entry" in SourceNamespaces of the IndicationFilter in the InteropNamespace. Deletion of an indication filter will also cause all subscriptions to that filter to be deleted. Deletion of a filter will not cause the deletion of any listener destination.

NOTE 7 Deleting the instance of an IndicationSubscription in the InteropNamespace will cause the corresponding IndicationSubscription in the "SourceNamespaces" to also be deleted. Deletion of an IndicationSubscription in a source namespace will not affect IndicationSubscriptions to filters in the Interop namespace (or other implementation namespaces). Deleting a subscription will not delete filters or listener destinations.

NOTE 8 Deleting the instance of ListenerDestinationCIMXML in the InteropNamespace will cause the corresponding instance (if one exists) to be deleted. Deleting an instance of ListenerDestinationCIMXML in any "source" namespace will not affect the corresponding instance in the InteropNamespace to be deleted.

EXPERIMENTAL

ModifyInstance - for IndicationFilters

```
void ModifyInstance (  
    [IN] <namedInstance> ModifiedInstance,  
    [IN, Optional, NULL] string propertyList[] = NULL  
)
```

The ModifiedInstance input parameter identifies the name of the Instance to be modified, and defines the set of changes to be made to the current Instance definition.

The only Property that may be specified in the PropertyList input parameter is the Query property. Modification of all other properties is not specified by SMI-S.

If successful, the specified Instance shall have been updated by the CIM Server.

If unsuccessful, one of the following status codes shall be returned by this method, where the first applicable error in the list (starting with the first element of the list, and working down) is

the error returned. Any additional method-specific interpretation of the error in is given in parentheses.

CIM_ERR_ACCESS_DENIED, CIM_ERR_NOT_SUPPORTED, CIM_ERR_INVALID_NAMESPACE, CIM_ERR_INVALID_PARAMETER (including missing, duplicate, unrecognized or otherwise incorrect parameters), CIM_ERR_INVALID_CLASS (the CIM Class of which this is to be a new Instance does not exist), CIM_ERR_NOT_FOUND (the CIM Instance does not exist), CIM_ERR_FAILED (some other unspecified error occurred).

EXPERIMENTAL

NOTE 9 Modifying the SourceNamespaces property of an IndicationFilter is not defined in this version of the International Standard. The results may vary depending on the implementation.

EXPERIMENTAL

43.6 Client Considerations and Recipes

43.6.1 Use of Profile Specific Recipes

This profile only defines the interfaces for creating indication filters, listener destinations and subscriptions to receive indications. For information related to indications defined by profiles, that information is document in the CIM Elements tables for the profile in question. For example, Array indications are documented in the CIM Elements table in the Array Profile (see *Information Technology – Storage Management, Part 4 Block Devices*, 5.8 “CIM Elements”).

43.6.2 General Client Considerations

The indication filters that a client subscribes to are either “predefined” and populated by the profile, or they are created by the client. If the profile supports “predefined” indication filters the client can find them via an enumeration. If the client cannot find the filter it is looking for, it may attempt to create the desired indication filter. If this fails, the client should fall back to creating a filter exactly as it exists in SMI-S. This shall work. The “predefined” indication filters in this International Standard shall be populated in the profile or it shall be possible to create it.

43.6.3 Discovery of Implementation variations

A client will need to discovery the variations that are allowed in SMI-S profile implementations. A profile implementation has the following degrees of variability:

- Client defined IndicationFilters, pre-defined IndicationFilters or both;
- InstModification, with or without PreviousInstance;
- Additional Indications.

To determine if an implementation supports Client Defined filters, the client should attempt to create an SMI-S specified filter. If it succeeds, the implementation supports client defined filters. At this point, the client can attempt to create a filter of its own choice or making (e.g., using the client’s desired query). If it fails, this means the implementation does not support an indication based on the query used.

If the attempt to create an SMI-S specified indication filter fails, this means client defined queries are not supported. At this point, the client should look for pre-defined filters. This can be done by enumerating filters in the namespace of the profile the client wishes to monitor.

An implementation may (or may not) support PreviousInstance, when the SMI-S International Standard for the profile identifies InstModification as the indication filter and PreviousInstance is identified as optional. If a client wishes to determine whether or not the implementation actually supports PreviousInstance, it can only tell by receiving an InstModification indication.

Additional Indications are IndicationFilters that are supported by the implementation, but not mandatory with SMI-S. If the implementation supports pre-defined Filters, these can easily be discovered in the enumeration of IndicationFilters. If the implementation does not support pre-defined filters, then the only way a client can discover these is through trial and error (or specific knowledge of the implementation).

43.6.4 Client Defined Filters

Clients need to avoid Filters that generate excessive events. Subscriptions to a general-purpose Server should be specific to the provider – for example “select * from CompanyCorp_InstCreation” rather than “select * from CIM_InstCreation”.

EXPERIMENTAL

43.6.5 Creation of IndicationFilter and ListenerDestination Instances

If an implementation populates scheme, hostname, and port in object path, the client should not change anything. If not, the client should accept namespace (and everything to the right) from the implementation and populate scheme, hostname, and port.

When creating IndicationFilters or ListenerDestination instances, clients should be prepared for CIM Servers expecting no key properties to be set and also be prepared for CIM Servers expecting all key properties to be set and valid. If key properties are specified, clients should be aware that the CIM Server may change the keys as specified in the CreateInstance request.

The recommended use of ElementName in these classes is that it is set by client to allow client to easily find instances in later client operations.

43.6.6 Creation of IndicationSubscription Instances

When creating IndicationSubscriptions, a client shall provide the references to the IndicationFilter and the ListenerDestination. But the client should not populate properties that it does not explicitly need to be set.

EXPERIMENTAL

43.6.7 Determine if the indication subscription requested already exists

```
// DESCRIPTION
// Determine if the indication subscription requested already exists. If
// not, then attempt to create the indication subscription passed in. If
// the CIM Server does not support the addition of indication, then the
// CIM Client will need to poll for these instance changes. This recipes
// does not handle the issue of providing the target URL for indications.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1.The namespace of interest has previously been identified and
//   defined in the #SomeNameSpace variable
// 2.The list of filters of interest has been previously built in the
//   #filters[] array. Each element in this array is the WQL filter
//   itself

// FUNCTION: createIndication
sub createIndication ($Filter)
{
    try {
```



```

        <create indications as per SMIS specification>
    } catch(CIM_ERROR_NOT_SUPPORTED) {
        <setup this class of instances to be polled for>
    }
}

// MAIN
$ExistingInstances[] = EnumerateInstances(#SomeNameSpace,
                                         "CIM_IndicationFilter")
for #i in $ExistingInstances
{
    for #j in #filters
    {
        if(!compare($ExistingInstances[#j].Query, #filters[#j]))
        {
            &createIndiciation(#filters[#j])
        }
    }
}

```

43.6.8 Listenable Instance Notification

```

// DESCRIPTION
// Create an indication subscription for every indication that is
// required by the profile.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1.The namespace of interest has previously been identified and
// defined in the #SomeNameSpace variable

#filters[] = <array of SMIS filters for the target profile>
@{Determine if Indications already exist or have to be created} #filters

```

43.6.9 Life Cycle Event Subscription Description

```

// DESCRIPTION
// Create an indication subscription for the operational status for a
// computer systems defined within a given CIM agent and namespace. This
// subscription will only be made in those CIM agents that have SAN
// devices or applications of interest defined in them. The client will
// have to determine once having received the indication, whether the
// computer system related to this indication (AlertingManagedElement
// attribute) is of interest. This recipe does not handle the target URL
// for the indication.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// None

#filter[0] = "SELECT * FROM CIM_InstModification
            WHERE SourceInstance ISA CIM_ComputerSystem
            AND SourceInstance.OperationalStatus[0] <>
            PreviousInstance.OperationalStatus[0]"
@{Determine if Indications already exist or have to be created} #filter

```


43.6.10 Subscription for alert indications

```
// DESCRIPTION
// Create an indication subscription for the alert indications defined
// within a given CIM agent and namespace. This subscription will only be
// made in those CIM agents that have SAN devices or applications of
// interest defined in them. The client will have to determine once
// having
// received the indication, whether the computer system related to this
// indication (AlertingManagedElement attribute) is of interest. Each
// specific alert indication will have also specific handling required
// for it by the CIM Client.
// NOTE: This recipe does not handle the target URL for the indication.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// None

#filter[0] = "SELECT * FROM CIM_AlertIndication"
@{Determine if Indications already exist or have to be created} #filter
```

43.6.11 Listenable Interface Modification Notification

```
// DESCRIPTION
// Create an indication subscription for every indication
// that is required by the profile
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1.The namespace of interest has previously been identified and
// defined in the #SomeNameSpace variable

#filters[] = <array of SMIS filters for the target profile>
@{Determine if Indications already exist or have to be created} #filters
```

43.6.12 Subscribe for Lifecycle Events where OperationalStatus Changes

```
// DESCRIPTION
// Create an indication subscription for the operational status for a
// computer systems defined within a given CIM agent and namespace. This
// subscription will only be made in those CIM agents that have SAN
// devices or applications of interest defined in them. The client will
// have to determine once having received the indication, whether the
// computer system related to this indication (AlertingManagedElement
// attribute) is of interest. This recipe does not handle the target URL
// for the indication.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// None

#filter[0] = "SELECT * FROM CIM_InstModification
WHERE SourceInstance ISA CIM_ComputerSystem
AND SourceInstance.OperationalStatus[0] <>
PreviousInstance.OperationalStatus[0]"
@{Determine if Indications already exist or have to be created} #filter
```

43.7 Registered Name and Version

Indication version 1.5.0 (Component Profile)

43.8 CIM Elements**43.8.1 General**

Table 416 describes the CIM elements for Indication.

Table 416 - CIM Elements for Indication

Element Name	Requirement	Description
43.8.2 CIM_AlertIndication	Optional	This Indication is used to capture events that occur in the profile, but may not be related to a specific part of the model.
43.8.3 CIM_IndicationFilter (client defined)	Optional	This is for 'client defined' CIM_IndicationFilter instances. CIM_IndicationFilter defines the value and format of an indication filter string.
43.8.4 CIM_IndicationFilter (pre-defined)	Optional	This is for 'pre-defined' CIM_IndicationFilter instances. CIM_IndicationFilter defines the value and format of an indication filter string.
43.8.5 CIM_IndicationSubscription	Mandatory	This association defines a subscription to a specific IndicationFilter instance by a specific indication handler (as represented by a ListenerDestinationCIMXML instance).
43.8.6 CIM_InstCreation	Optional	CIM_InstCreation is an indication of the creation of a CIM instance. It would be generated when an instance of the SourceInstance class is created (either explicitly or implicitly).
43.8.7 CIM_InstDeletion	Optional	CIM_InstDeletion is an indication of the Deletion of a CIM instance. It would be generated when an instance of the SourceInstance class is deleted from the model (either explicitly or implicitly).
43.8.8 CIM_InstModification	Optional	CIM_InstModification is an indication of the modification or change to a CIM instance. It would be generated when an instance of the SourceInstance class is modified or changed (either explicitly or implicitly).
43.8.9 CIM_ListenerDestinationCIMXML (Indication Handler)	Mandatory	A CIM_ListenerDestinationCIMXML describes the destination for CIM Export Messages to be delivered via CIM-XML. ListenerDestinationCIMXML is subclassed from ListenerDestination.

43.8.2 CIM_AlertIndication

A CIM_AlertIndication is a specialized type of CIM_Indication that contains information about the severity, cause, recommended actions and other data of a real world event.

CIM_AlertIndication is subclassed from CIM_ProcessIndication.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 417 describes class CIM_AlertIndication.

Table 417 - SMI Referenced Properties/Methods for CIM_AlertIndication

Properties	Flags	Requirement	Description and Notes
IndicationIdentifier		Optional	An identifier for the Indication used for correlated indications.
CorrelatedIndications		Optional	IndicationIdentifiers whose notifications are correlated with this one.
IndicationTime	N	Mandatory	The time and date of creation of the Indication. The property may be set to NULL if it cannot be determined.
Description		Optional	A free form text description.
AlertingManagedElement		Mandatory	The identifying information of the entity for which this Indication is generated.
AlertingElementFormat		Mandatory	Valid SMI-S values are 0 1 2 ('Unknown' 'Other' 'CIMObjectPath').
AlertType		Mandatory	This shall be 1 2 3 4 5 6 7 8 ('Other' 'Communications Alert' 'Quality of Service Alert' 'Processing Error' 'Device Alert' 'Environmental Alert' 'Model Change' 'Security Alert').
OtherAlertType		Optional	
PerceivedSeverity		Mandatory	This shall be 0 1 2 3 4 5 6 7 ('Unknown', 'Other' 'Information' 'Degraded/Warning' 'Minor' 'Major' 'Critical' 'Fatal/NonRecoverable').
OtherSeverity		Optional	
ProbableCause		Mandatory	Many possible values in a value map. See MOF.
ProbableCauseDescription		Optional	
EventID		Optional	
SystemCreationClassName		Mandatory	
SystemName		Mandatory	The scoping System's Name for the Provider generating this Indication. The SystemName would typically be the name of the system that generates the indication.
ProviderName		Mandatory	
OwningEntity	N	Optional	A string that uniquely identifies the entity that owns the definition of the format of the Message.
MessageID	N	Optional	A string that uniquely identifies, within the scope of the OwningEntity, the format of the Message.
Message	N	Optional	The formatted message (including the MessageArguments).
MessageArguments	N	Optional	An array of strings that contain the dynamic content of the message.
OtherAlertingElementFormat	N	Optional	Not Specified in this version of the Profile.

Table 417 - SMI Referenced Properties/Methods for CIM_AlertIndication

Properties	Flags	Requirement	Description and Notes
Trending	N	Optional	Not Specified in this version of the Profile.
RecommendedActions	N	Optional	Not Specified in this version of the Profile.
EventTime	N	Optional	Not Specified in this version of the Profile.

43.8.3 CIM_IndicationFilter (client defined)

CIM_IndicationFilter instances that are 'client defined' are IndicationFilters that are created by a client using CreateInstance. If a profile implementation can support client defined IndicationFilters, the implementation would support 'client defined' IndicationFilter instances. The implementation shall support 'client defined' filters that are defined by SMI-S profile as mandatory, but may also support additional filters supported by the implementation.

CIM_IndicationFilter is subclassed from CIM_ManagedElement.

Created By: CreateInstance

Modified By: ModifyInstance

Deleted By: DeleteInstance

Requirement: Optional.

Table 418 describes class CIM_IndicationFilter (client defined).

Table 418 - SMI Referenced Properties/Methods for CIM_IndicationFilter (client defined)

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
Name		Mandatory	This should take the form OrgID ":" RegisteredName ":" UniqueID. For more details, see section <i>Information Technology – Storage Management, Part 3 Common Profiles</i> 43.1.2 IndicationFilter Names.
SourceNamespace	N	Optional	Deprecated. For instances in the InteropNamespace, this shall be the namespace where the indications are to originate. For instances in the implementation namespace where the indications are to originate (e.g., the namespace of the profile that supports the filter), this may be NULL to indicate the Filter is registered in the Namespace where the indications originate.
SourceNamespaces	N	Mandatory	This should be all the namespaces where the indications may originate.
Query		Mandatory	A string that specifies (in QueryLanguage terms) which indications are to be delivered to the ListenerDestinations.
QueryLanguage		Mandatory	This shall be 'DMTF:CQL' for CQL queries, but may be 'WQL' or 'SMI-S V1.0'. WQL and SMI-S V1.0 are deprecated in favor of 'DMTF:CQL'.

Table 418 - SMI Referenced Properties/Methods for CIM_IndicationFilter (client defined)

Properties	Flags	Requirement	Description and Notes
ElementName		Optional	A Client Defined user friendly string that identifies the Indication Filter.
Caption	N	Optional	Not Specified in this version of the Profile.
Description	N	Optional	Not Specified in this version of the Profile.

43.8.4 CIM_IndicationFilter (pre-defined)

CIM_IndicationFilter instances that are 'pre-defined' are IndicationFilters that are populated automatically by the profile provider. If a profile implementation cannot support client defined IndicationFilters, the implementation can populate its model with 'pre-defined' IndicationFilter instances. 'Pre-defined' filters shall include those that are required by the profile, but may also contain additional filters supported by the implementation.

CIM_IndicationFilter is subclassed from CIM_ManagedElement.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 419 describes class CIM_IndicationFilter (pre-defined).

Table 419 - SMI Referenced Properties/Methods for CIM_IndicationFilter (pre-defined)

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
Name		Mandatory	This should take the form OrgID ":" RegisteredName ":" UniqueID. For more details, see section <i>Information Technology – Storage Management, Part 3 Common Profiles</i> 43.1.2 IndicationFilter Names.
SourceNamespace	N	Optional	Deprecated. For instances in the InteropNamespace, this shall be the namespace where the indications are to originate. For instances in the implementation namespace where the indications are to originate (e.g., the namespace of the profile that supports the filter), this may be NULL to indicate the Filter is registered in the Namespace where the indications originate.
SourceNamespaces	N	Mandatory	This should be all the namespaces where the indications may originate.
Query		Mandatory	A string that specifies (in QueryLanguage terms) which indications are to be delivered to the ListenerDestinations.
QueryLanguage		Mandatory	This shall be 'DMTF:CQL' for CQL queries, but may be 'WQL' or 'SMI-S V1.0'. WQL and SMI-S V1.0 are deprecated in favor of 'DMTF:CQL'.

Table 419 - SMI Referenced Properties/Methods for CIM_IndicationFilter (pre-defined)

Properties	Flags	Requirement	Description and Notes
ElementName	N	Optional	SMI-S does not specify this property for pre-defined IndicationFilters.
Caption	N	Optional	Not Specified in this version of the Profile.
Description	N	Optional	Not Specified in this version of the Profile.

43.8.5 CIM_IndicationSubscription

A CIM_IndicationSubscription is not subclassed from anything.

Created By: CreateInstance

Modified By: Static

Deleted By: DeleteInstance

Requirement: Mandatory.

Table 420 describes class CIM_IndicationSubscription.

Table 420 - SMI Referenced Properties/Methods for CIM_IndicationSubscription

Properties	Flags	Requirement	Description and Notes
RepeatNotificationPolicy		Mandatory	SMI-S supports a restricted set of values. This shall be 2 3 4 ('None' 'Suppress' 'Delay').
RepeatNotificationInterval		Optional	Mandatory if the RepeatNotificationPolicy is 'Suppress' or 'Delay'.
RepeatNotificationGap		Optional	Mandatory if the RepeatNotificationPolicy is 'Delay'.
RepeatNotificationCount		Optional	Mandatory if the RepeatNotificationPolicy is 'Suppress' or 'Delay'.
LastIndicationIdentifier		Optional	The IndicationIdentifier of the last indication produced for this subscription regardless if that indication were delivered.
LastIndicationProductionDateTime		Optional	The date and time of the production of the last indication produced for this subscription regardless if that indication were delivered.
OnFatalErrorPolicy	N	Optional	Not Specified in this version of the Profile.
OtherOnFatalErrorPolicy	N	Optional	Not Specified in this version of the Profile.
FailureTriggerTimeInterval	N	Optional	Not Specified in this version of the Profile.
SubscriptionState	N	Optional	Not Specified in this version of the Profile.
OtherSubscriptionState	N	Optional	Not Specified in this version of the Profile.
TimeOfLastStateChange	N	Optional	Not Specified in this version of the Profile.
SubscriptionDuration	N	Optional	Not Specified in this version of the Profile.
SubscriptionStartTime	N	Optional	Not Specified in this version of the Profile.
SubscriptionTimeRemaining	N	Optional	Not Specified in this version of the Profile.
OtherRepeatNotificationPolicy	N	Optional	Not Specified in this version of the Profile.

Table 420 - SMI Referenced Properties/Methods for CIM_IndicationSubscription

Properties	Flags	Requirement	Description and Notes
AlertOnStateChange	N	Optional	Not Specified in this version of the Profile.
Filter		Mandatory	
Handler		Mandatory	

43.8.6 CIM_InstCreation

CIM_InstCreation notifies a handler when a new instance (of a class defined in the Filter QueryString) is created.

CIM_InstCreation is subclassed from CIM_InstIndication.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 421 describes class CIM_InstCreation.

Table 421 - SMI Referenced Properties/Methods for CIM_InstCreation

Properties	Flags	Requirement	Description and Notes
IndicationIdentifier		Optional	An identifier for the Indication used for correlated indications.
CorrelatedIndications		Optional	IndicationIdentifiers whose notifications are correlated with this one.
IndicationTime		Mandatory	The time and date of creation of the Indication. The property may be set to NULL if it cannot be determined.
SourceInstance		Mandatory	A copy of the instance that changed to generate the Indication. SourceInstance contains the current values of the properties selected by the Indication Filter's Query.
SourceInstanceModelPath		Mandatory	The Model Path of the SourceInstance.
PerceivedSeverity	N	Optional	Not Specified in this version of the Profile.
OtherSeverity	N	Optional	Not Specified in this version of the Profile.
SourceInstanceHost	N	Optional	Not Specified in this version of the Profile.

43.8.7 CIM_InstDeletion

CIM_InstDeletion notifies a handler when a new instance (of a class defined in the Filter QueryString) is deleted.

CIM_InstDeletion is subclassed from CIM_InstIndication.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 422 describes class CIM_InstDeletion.

Table 422 - SMI Referenced Properties/Methods for CIM_InstDeletion

Properties	Flags	Requirement	Description and Notes
IndicationIdentifier		Optional	An identifier for the Indication used for correlated indications.
CorrelatedIndications		Optional	IndicationIdentifiers whose notifications are correlated with this one.
IndicationTime		Mandatory	The time and date of creation of the Indication. The property may be set to NULL if it cannot be determined.
SourceInstance		Mandatory	A copy of the instance that changed to generate the Indication. SourceInstance contains the current values of the properties selected by the Indication Filter's Query.
SourceInstanceModelPath		Mandatory	The Model Path of the SourceInstance.
PerceivedSeverity	N	Optional	Not Specified in this version of the Profile.
OtherSeverity	N	Optional	Not Specified in this version of the Profile.
SourceInstanceHost	N	Optional	Not Specified in this version of the Profile.

43.8.8 CIM_InstModification

CIM_InstModification notifies a handler when a new instance (of a class defined in the Filter QueryString) is modified or changed. To avoid undue effort on Providers, the select list (in the query filter) for this indication should only call for properties that are needed.

CIM_InstModification is subclassed from CIM_InstIndication.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 423 describes class CIM_InstModification.

Table 423 - SMI Referenced Properties/Methods for CIM_InstModification

Properties	Flags	Requirement	Description and Notes
IndicationIdentifier		Optional	An identifier for the Indication used for correlated indications.
CorrelatedIndications		Optional	IndicationIdentifiers whose notifications are correlated with this one.
IndicationTime		Mandatory	The time and date of creation of the Indication. The property may be set to NULL if it cannot be determined.
SourceInstance		Mandatory	A copy of the instance that changed to generate the Indication. SourceInstance contains the current values of the properties selected by the Indication Filter's Query.
SourceInstanceModelPath		Mandatory	The Model Path of the SourceInstance.

Table 423 - SMI Referenced Properties/Methods for CIM_InstModification

Properties	Flags	Requirement	Description and Notes
PreviousInstance		Optional	A copy of the 'previous' instance whose change generated the Indication. PreviousInstance contains 'older' values of an instance's properties (as compared to SourceInstance), selected by the IndicationFilter's Query.
PerceivedSeverity	N	Optional	Not Specified in this version of the Profile.
OtherSeverity	N	Optional	Not Specified in this version of the Profile.
SourceInstanceHost	N	Optional	Not Specified in this version of the Profile.

43.8.9 CIM_ListenerDestinationCIMXML (Indication Handler)

CIM_ListenerDestinationCIMXML is subclassed from CIM_ListenerDestination.

Created By: CreateInstance

Modified By: Static

Deleted By: DeleteInstance

Requirement: Mandatory

Table 424 describes class CIM_ListenerDestinationCIMXML (Indication Handler).

Table 424 - SMI Referenced Properties/Methods for CIM_ListenerDestinationCIMXML (Indication Handler)

Properties	Flags	Requirement	Description and Notes
ElementName		Mandatory	A client defined user friendly string that identifies the CIMXML Listener destination.
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	
PersistenceType		Mandatory	For SMI-S, this shall be 2 3 ('permanent' 'transient').
Destination		Mandatory	The destination URL to which CIM-XML Export Messages are to be delivered. The scheme prefix shall be consistent with the DMTF CIM-XML specifications. If a scheme prefix is not specified, the scheme 'http:' shall be assumed.
Caption	N	Optional	Not Specified in this version of the Profile.
Description	N	Optional	Not Specified in this version of the Profile.
OtherPersistenceType	N	Optional	Not Specified in this version of the Profile.

STABLE

EXPERIMENTAL

44 Experimental Indication Profile

44.1 Description

44.1.1 General

The Experimental Indication Profile supports the Indication Profile and extends it with the following experimental content:

- Client Certificates for securing indications delivery
- Use of a Typed WBEM URI for identifying the AlertingManagedElement
- Scheme for encoding of the IndicationIdentifier
- The use of CQL as the Query Language for IndicationFilters
- Use of Bellwether events to control indication storms
- Use of batching to control indication storms
- Use and encoding of CorrelatedIndications
- Semi-fixed Client Specific Indications
- Filter Collections (both predefined and client defined)
- Indication Configuration Service (extrinsic methods)

IECNORM.COM : Click to view the full PDF of ISO/IEC 24775-3:2014

44.1.2 Basic Indication Classes and Association

Figure 60 illustrates the classes used in support of indications.

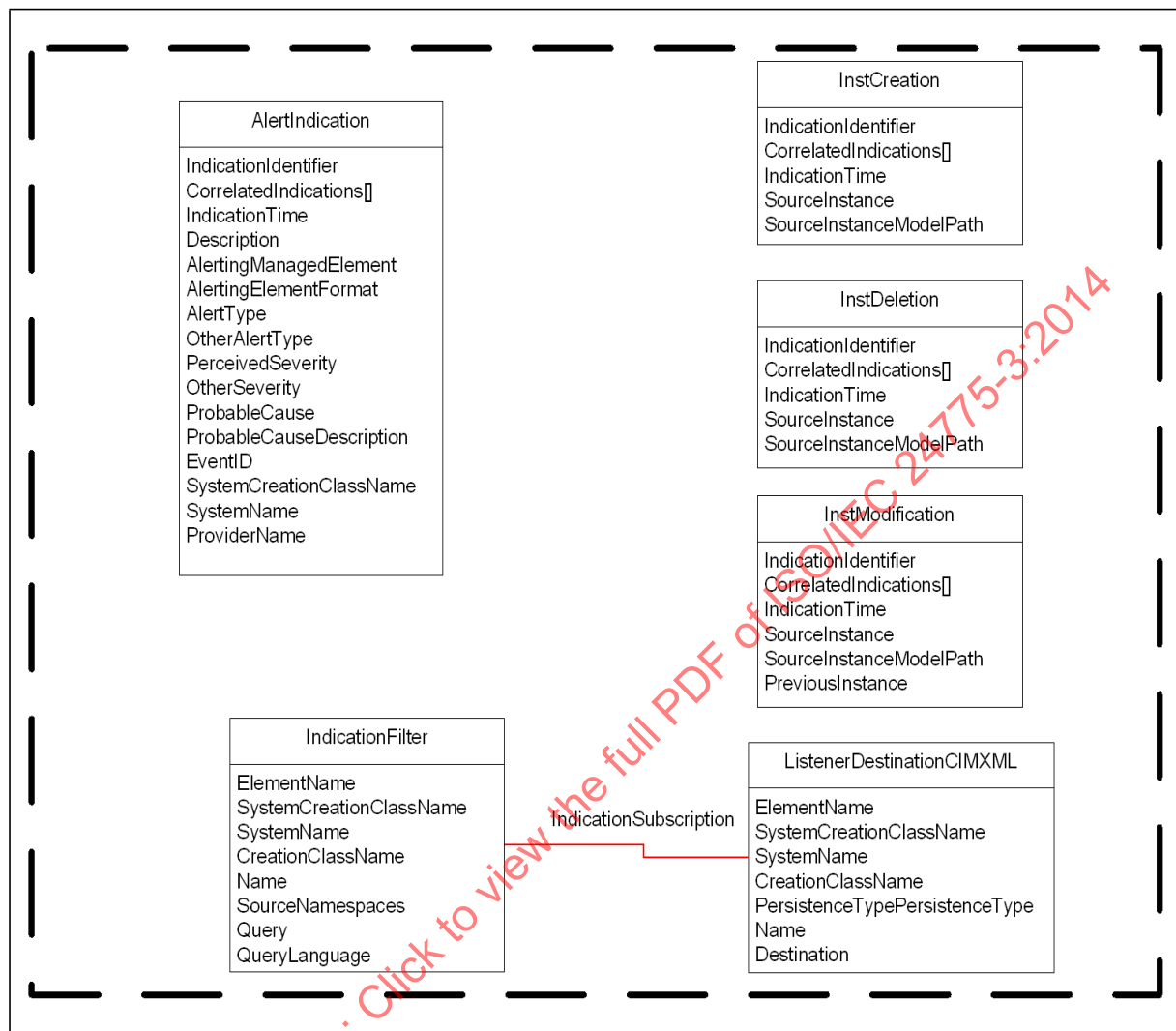


Figure 60 - Indication Profile Instance Diagram

44.1.3 AlertingManagedElement encoding in AlertIndication Instances

When encoding the mandatory property “AlertingManagedElement” of an AlertIndication the following rules apply:

- If the element in question is modeled by the profile implementation, then the format for this property should be as a Typed WBEM URI as defined in DSP0207.
- If the element in question is not modeled by the profile implementation, then the encoding for this property should be as meaningful to clients as possible

44.1.4 Instrumentation Requirements

EXPERIMENTAL

44.1.4.1 Indication Identification

Indications are identified through the IndicationIdentifier property. An indication can be correlated to previously produced indications through the use of the CorrelatedIndication property. Generally, the identity of the indication is only meaningful as a correlatable ID within

the CorrelatedIndication property or in its relevancy to the LastIndicationIdentifier property in the IndicationSubscription class.

The LastIndicationIdentifier property on the IndicationSubscription association should record the identity of the last indication produced for the combination of IndicationFilter and IndicationDestination that the association instance represents.

NOTE 1 The LastIndicationIdentifier property will become mandatory in a future release of SMI-S as WBEM infrastructures are enhanced to support the property.

The client can determine if it did get delivery of any indication destined for it by comparing the last indication it received, or the last indication it received for a particular indication subscription, with the LastIndicationIdentifier. It is important for clients to be able to determine if there are interruptions in the indication telemetry. Confidence in the indication delivery combined with the ability to determine the extent of the failure to receive indications, provides clients with a mechanism to gauge appropriately the response to the failures and avoid having to flush state and explore the SMI-S Server's model again.

NOTE 2 In a future release of SMI-S, the modeling of the health of the indication delivery system or service will help clients to determine if there are problems in the configuration of the subscription and related credentials, or in the indication delivery configuration of the SMI-S Server. This design will require the logging of the errors produced in the delivery of the indications.

The naming algorithm for the IndicationIdentifier property, shown in Figure 61, includes the population of the two subcomponents of the property, OrgID and LocalID, as separated by a colon ":". The OrgID shall contain a registered trademark for the developer of the implementation producing the indication. The LocalID shall contain the combination of the CIM Object Name of the IndicationFilter that produced the Indication, production sequence number, and a delivery sequence number. These sequence numbers are in the form of an unsigned integer. These three elements within the LocalID are separated by a hash "#". The omission of the Handler key property of the indication subscription, which is the object name of the indication destination, means that the client should assume that the indication was correctly delivered to it.

```
<Trademark>:<Object Name of IndicationFilter>#<production sequence number>#<delivery sequence number>
```

Figure 61 - Anatomy of IndicationIdentifier

The production sequence number is a count of all indications produced by this SMI-S implementation. The sequence shall be unique by device or application instrumented through SMI-S. The production sequence number shall not be unique by indication filter, but instead shall represent the count of indications produced for this device or application. Any gaps in the production sequence number represent indications that were produced but were not delivered because there are no indications' subscribers or a SMI-S Client did not receive the indication because it was not subscribed to it.

The delivery sequence number range shall be unique and independent by indication subscription. The delivery sequence number reported shall increase by one and only one with every indication produced for that subscription. In other words, this delivery sequence number can be viewed as a count of indications produced for a particular indication subscription. Any gaps in the delivery sequence number represent indications that were produced for a particular destination (e.g., a SMI-S Client) but were not delivered for some reason. Since an SMI-S Server, the infrastructure, is normally in charge of forwarding indications delivered to it by CIM Providers, it is best able to produce this sequence number. SMI-S Servers should produce this sequence number, but may omit it if unsupported by the CIM Server.

It is recommend that the sequence numbers have a 16-bit range. In other words, the sequence numbers should start at 1 and iterate to 65,536. The implementation may use a larger range, like 1 to 262,144, and should do so if there is a possibility that 65,536 indications per a given

indication subscription can be produced within a twenty-four hour period. Regardless, the maximum sequence number shall be a power of two.

The implementation may roll-over the sequence number and start again at one. The requirement that the sequence number shall be a power of two allows a client to determine what the maximum the sequence could be, like 65,536, in order to determine if the last sequence number received for an indication subscription, e.g., 65,533, is the last one it should have received. Clients can not be certain how many indications were missed when the sequence number rolls over given the unknown frequency of indication production and the unknown maximum value of the sequence.

Conformance to the indication identifier naming algorithm is mandatory.

However, an indicator that many indications for a given subscription may have been missed is contained in the `LastIndicationProductionDateTime` property. The difference between now and the date and time value of `LastIndicationProductionDateTime` is significant, then the possibility exists.

EXPERIMENTAL

44.1.4.2 Handling of Indication Storms

44.1.4.2.1 General

To contain the impact of indication storms an implementation can employ additional techniques:

- Use of Bellwether events (if they are defined by the profile).

44.1.4.2.2 Use of Bellwether Events

There are many state changes in the model for a device or application that results in changes in many CIM instances. For example, the addition of a device or application representation to a CIMOM should result in creation indications for every single member instance of that device or application. The activation of a `ZoneSet` from one of the member `Switches` in a fabric should result to indication listeners on another `Switch`'s namespace creation indications for every instance of the new `ZoneSet`.

The worse case risk is that several of this type of situation may occur simultaneously and result in network storms and the sudden saturation of the LAN. Additionally, the use of computing resources of the device or application producing the indication or client receiving the indications may be unacceptably high.

Indications provide the most value when they are used by a client as a mechanism to pick a significant or small number of changes in CIMOMs of interest. In order to capture a wide variety of changes, any of which may be pertinent to the client application, the client is likely to create many indication subscriptions and keep them all active simultaneously. This approach is not problematic because the number of management related changes to any device or application in the network is usually very small.

As mentioned previously, there are several potential situations where an excessive number of indications can be produced, thereby potentially overloading the network, originating CIMOM, and receiving client's resources. There is no need to occur such a risk because it is likely that the client is not going to be interested in all things at all times. The interest of the client in instance changes usually follows the needs of the current users of that client application.

Bellwether indications are used by SMI-S designers and individual implementation to signal many instance changes with one event. A client can assume that some previously defined graph of associated CIM instances are affected when it receives a bellwether indication. It can then choose, if warranted, to fetch all or some of these instances. This design prevents the previously mentioned adverse side effects.

Some rules being considered are:

- When a device or application is added to a namespace and there are indication subscription that cover some or all of the graph of instances added by side effect of the addition, then only a create indication is produced for the top level object for the device or application, like ComputerSystem, provided that there is an indication subscription for changes in the top-level object. Similarly, if a device or application is deleted in the same situation, then only a delete indication will be produced.
- Bellwether indication are mandatory if they exist in SMI-S and will be easily identified as being bellwether events.
 - The classes associated to the bellwether indication will be part of the definition of the indication. The client can assume that instances of these classes will have been affected and can choose to harvest that data. The implementation is not required to produce instances of every class listed as per the requirements defined elsewhere in SMI-S.
- SMI-S Designer's are encouraged to define bellwether indications, which can be of any class of indication, for major state changes of a model. In the previous examples, the device creation could be a life cycle indication where changes in ZoneSet change may be best communicated by an Alert Indication.

44.1.4.2.3 Bellwether Indications for ComputerSystem

It is important to not overload a SMI-S client when device or applications are added or removed from CIM Object Managers. The addition or removal of the representation of a device or application is attributed to the creation or deletion of a top-level computer system instance. This overloading would arise from a SMI-S Agent sending creation or deletion indications to every indication destination for all component or dependent instances to the top-level computer system. For this profile, when a top-level computer system instance is created in the model, the SMI-S agent shall not produce indications for indication subscriptions, on indications that do not reference the top-level computer system, that would otherwise receive InstCreation indications. Likewise, for this profile, when a top-level computer system is deleted from the model, the SMI-S agent shall not produce indications for indication subscriptions, on indications that do not reference the top-level computer system, that would otherwise receive InstDeletion indications.

44.1.5 Semi-Fixed Client Specific Indication Filters

44.1.5.1 General

Semi-fixed Client specific IndicationFilters extend the support for indications in the following classes:

- SNIA_IndicationFilterTemplate

This class mirrors the CIM_IndicationFilter, but the Query property supports a query with the string 'SUBSTITUTION_STRING' included in a CQL query. This is a template that may be used by a client application to create an instance of CIM_IndicationFilter with a client application supplied string in place of the string 'SUBSTITUTION_STRING'.

- SNIA_IndicationConfigurationCapabilities (and the SupportedFeatures property)

The SupportedFeatures property of SNIA_IndicationConfigurationCapabilities includes the value '7' ("Semi-fixed IndicationFilters") that indicates semi-fixed IndicationFilters are supported.

When an implementation sets this enumeration, the implementation shall support creation (CreateInstance) of IndicationFilters that follow a pattern that includes substitution strings as defined in the

SNIA_IndicationFilterTemplate Query property. The client is allowed to replace the substitution string with any simple expression (including constants).

Semi-fixed IndicationFilterTemplates are documented in this International Standard with the substitution string identified with the string 'SUBSTITUTION_STRING' in the Query property. For example, the following Query values would indicate a Semi-fixed IndicationFilterTemplate:

```
SELECT * FROM CIM_InstDeletion
WHERE SourceInstance ISA CIM_StorageSynchronized AND
OBJECTPATH(SourceInstanceModelpath) = OBJECTPATH('SUBSTITUTION_STRING')
```

or

```
SELECT * FROM CIM_InstModification
WHERE SourceInstance ISA CIM_StorageVolume AND
SourceInstance.OperationalStatus = 'SUBSTITUTION_STRING'
```

When a semi-fixed IndicationFilterTemplate is defined in this International Standard, the description column for the CIM Element for the Indication will identify valid substitutions or will reference an implementation section that identifies the valid values.

44.1.5.2 Naming Conventions for IndicationFilterTemplates and IndicationFilters

Both IndicationFilters and IndicationFilterTemplates have a Name property. The value of the Name property shall be formatted as defined by the following ABNF rule:

```
OrgID ":" RegisteredName ":" UniqueID
```

Where OrgID identify the business entity owning the referencing profile. OrgID shall include a copyrighted, trademarked, or otherwise unique name that is owned by that business entity or that is a registered ID assigned to that business entity by a recognized global authority. In addition, to ensure uniqueness, OrgID shall not contain a colon (:).

For referencing profiles owned by the SNIA, OrgID shall match "SNIA" for IndicationFilterTemplates defined by the standard. For vendor unique IndicationFilterTemplates, the OrgID should be a unique name for the vendor. For client defined IndicationFilters that are based on IndicationFilterTemplates, the OrgID should identify the client (application) organization.

The RegisteredName shall be the registered name of the referencing profile, as defined by the value of its CIM_RegisteredProfile.RegisteredName property.

The UniqueID shall uniquely identify the instance within the referencing profile.

44.1.6 Filter Collections

44.1.6.1 General

The International Standard supports two types of filter collections:

- **Predefined Filter Collections** - The predefined filter collections augments the Indication Profile support for predefined indication filters and indication filter templates by collecting them into a structure of collections (one per profile) that are hosted on the top level system of the autonomous profile. This provides a convenient means of finding an implementations support for predefined filters.
- **Client Defined Filter Collections** - The client defined filter collections are collections that are defined by applications of the standard. Client defined filter collections may include predefined and/or client defined indication filters and allow the application to collect a set of related indication filters to which the application wishes to subscribe.

44.1.6.2 Predefined Filter Collections

Predefined filter collections are an optional feature of the International Standard. Support would be indicated via the IndicationConfigurationCapabilities (see 44.1.7) in the SupportedFeatures property. If the SupportedFeatures array includes the value '5' it means the implementation supports predefined filter collections.

Figure 62 illustrates the classes associated with predefined FilterCollection support.

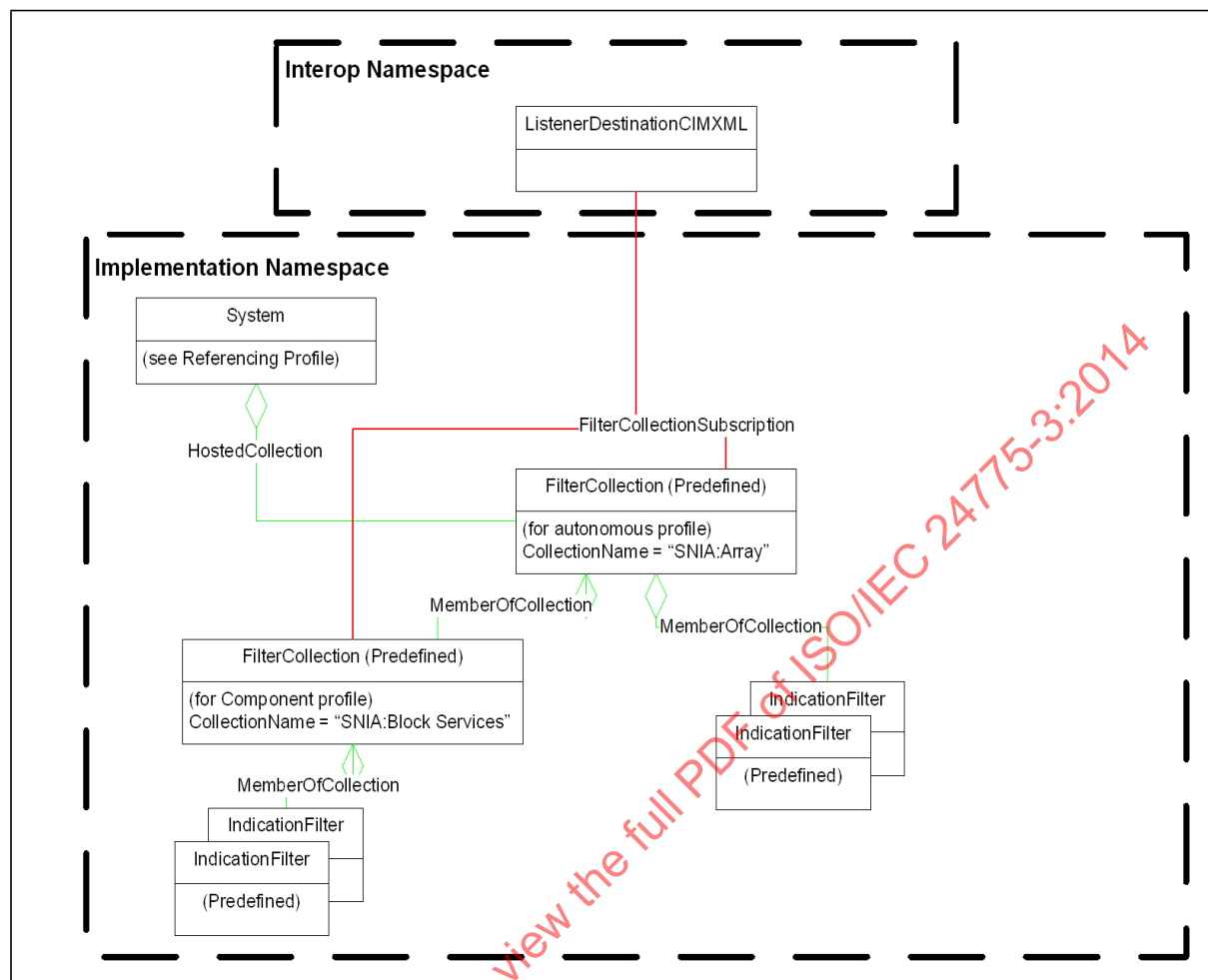


Figure 62 - Predefined Filter Collections

Support for predefined filter collections includes instantiations of the following classes and associations to the model:

- `FilterCollection (Predefined)`

A predefined `FilterCollection` collects a set of predefined `IndicationFilters`. The primary purpose of a predefined `FilterCollection` is for an implementation to declare the `IndicationFilters` that it supports. Minimally this should include all `IndicationFilters` that are defined as mandatory for the profile. However, it may also include optional, conditional or vendor extension `IndicationFilters` supported by the implementation.

One predefined `FilterCollection` is defined for each profile supported by the implementation. The `FilterCollections` are organized in a 2 level hierarchy. The top most `FilterCollection` is the `FilterCollection` for the autonomous profile. The name of the `FilterCollection` (`CollectionName` property) is of the form "SNIA:<profile name>". In Figure 62 the autonomous profile is an Array profile. In addition to collecting predefined `IndicationFilters` of the autonomous profile, the top level `FilterCollection` would also collect `FilterCollections` for each of the component profiles supported by the in implementation. In Figure 62 the component profile `FilterCollection` shown is for the Block Services Package (`CollectionName = "SNIA:Block Services"`).

- `HostedCollection`

This associates each filter collection with the top level system for which the collection applies (e.g., the top level system of the autonomous profile).

- FilterCollectionSubscription

This associates (subscribes) a ListenerDestination (Handler) to a FilterCollection (collection). All indication filters in the collection will be reported to the ListenerDestination (without the need for individual subscriptions on the indication filters).

Note that a FilterCollectionSubscription will not return any indications for any IndicationFilterTemplates in a pre-defined FilterCollection. Templates are only used for creation of client defined IndicationFilters.

A client may subscribe to the collection of indications rather than subscribing to each individual filter in the collection. However, a client might prefer to form its own collection of filters that it wants to subscribe to (see 44.1.6.3).

- MemberOfCollection

This associates predefined IndicationFilters and predefined component FilterCollections to the FilterCollections in which they belong. MemberOfCollection associations are static and established by the implementation.

44.1.6.3 Client Defined Filter Collections

Client defined filter collections are an optional feature of the International Standard. Support would be indicated via the IndicationConfigurationCapabilities (see 44.1.7) in the SupportedFeatures property. If the SupportedFeatures array includes the value '6' it means the implementation supports client defined filter collections.

Figure 63 illustrates the classes associated with client defined FilterCollection support.

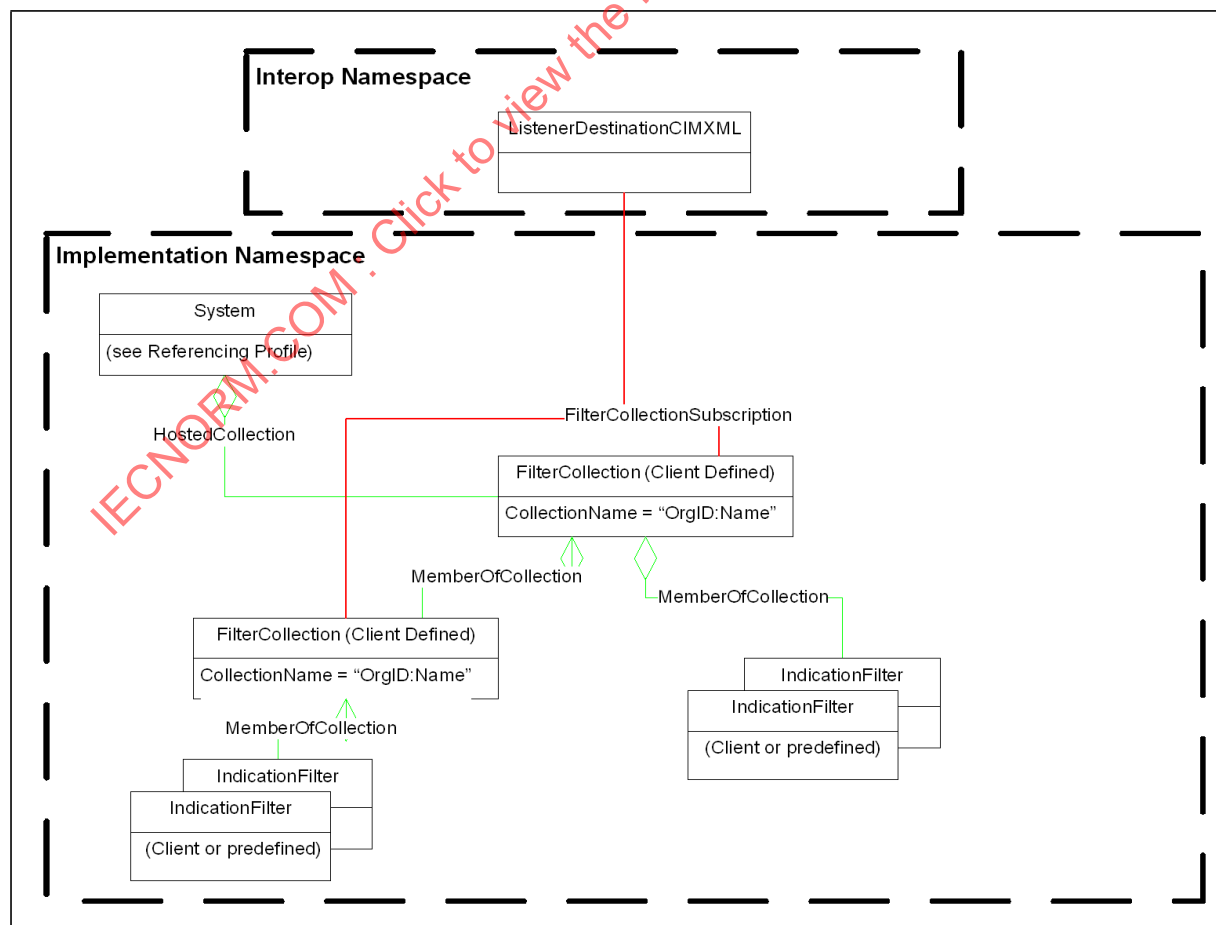


Figure 63 - Client Defined Filter Collections

Support for client defined filter collections includes support of the following classes and associations to the model:

- FilterCollection (Client Defined)

A client defined FilterCollection collects a set of IndicationFilters (or other FilterCollections). The primary purpose of a client defined FilterCollection is to allow a client to establish a set of indication filters in which it wishes to subscribe to as a group. The indication filters collected may be either client defined or predefined. They may include any IndicationFilters supported by the implementation.

Unlike predefined FilterCollections client defined FilterCollections may be organized for the convenience of the client application. The FilterCollections may be organized in a hierarchy of any number of levels and a any one FilterCollection need not correspond to a profile.

The name of the FilterCollection (CollectionName property) is of the form "<OrgID:<Unique Name>". In Figure 63 a top level collection is defined with a lower level FilterCollection. The <OrgID> component should be a company indicator (e.g., stock ticker). The <Unique Name> part of the name should uniquely identify the collection within that company. It may be desirable to make the <Unique Name> part of the CollectionName a compound construction (e.g., <Product Name:Name within Product>). But all that this standard dictates is that the <OrgID> cannot be "SNIA".

A set of methods for creating and maintaining client defined FilterCollections are provided by the IndicationConfigurationService (see 44.1.7).

- HostedCollection

This associates each filter collection with the top level system for which the collection applies (e.g., the top level system of the autonomous profile).

- FilterCollectionSubscription

This associates (subscribes) a ListenerDestination (Handler) to a FilterCollection (collection). All indication filters in the collection will be reported to the ListenerDestination (without the need for individual subscriptions on the indication filters).

A client may subscribe to the collection of indications rather than subscribing to each individual filter in the collection.

- MemberOfCollection

This associates IndicationFilters (predefined or client defined) and other FilterCollections (client defined or predefined) to higher level FilterCollections. MemberOfCollection is established using methods of the IndicationConfigurationService (see 44.1.7).

44.1.7 Indication Configuration Services

Figure 64 illustrates the classes associated with support for methods for configuring and testing indications support.

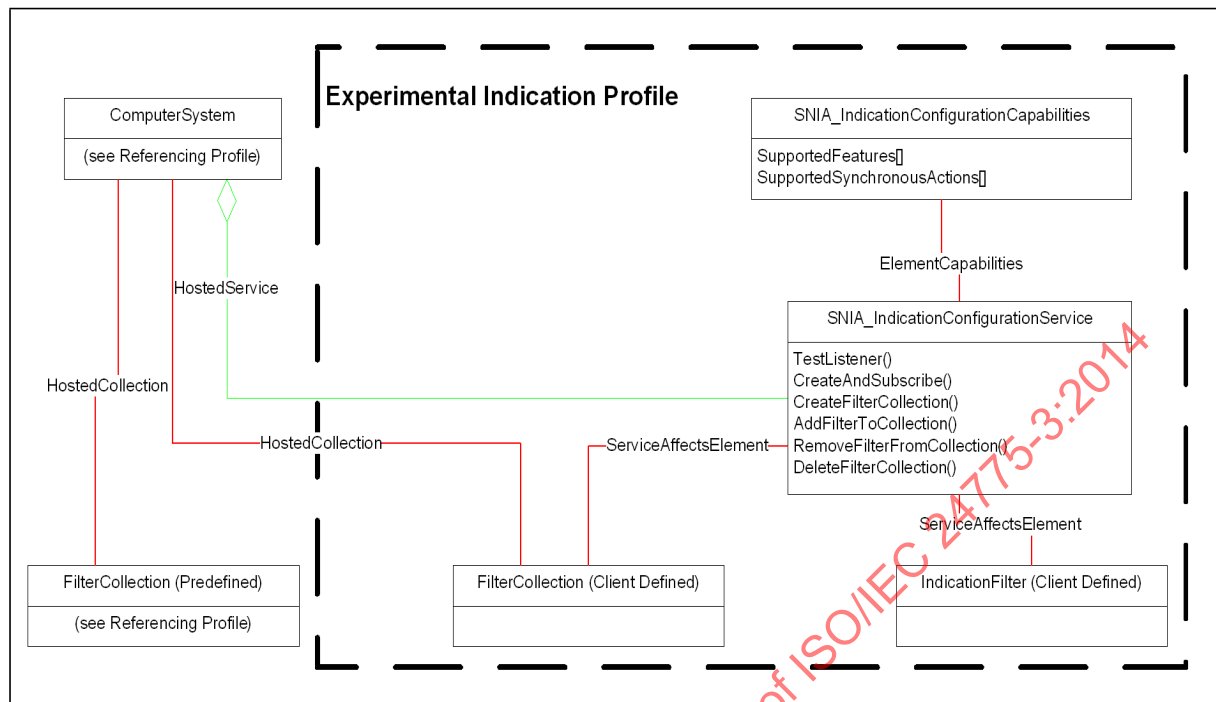


Figure 64 - Indication Configuration Service Classes

Support for the Indication Configuration Service add the following classes and associations to the model:

- SNIA_IndicationConfigurationService

This service includes methods for testing a listener, handling the creation and subscription to indications as one extrinsic method (rather than a set of CreateInstances) and methods for managing client defined filter collections. The SNIA_IndicationConfigurationService is a service that is specific to a particular profile implementation and the classes managed are instantiated in the implementation namespace. The SNIA_IndicationConfigurationService shall be instantiated when the Experimental Indication Profile is supported, but support for the individual methods are conditional (See the SNIA_IndicationConfigurationCapabilities, 44.8.17).

Note that predefined FilterCollections are not managed by the methods of the indication configuration services. They are, by their nature, static and maintained by the implementation.

- HostedService

This associates the SNIA_IndicationConfigurationService to the system of the autonomous profile (referencing profile) for which the service applies. For example, for the Array Profile, the indication configuration service would be hosted on the top level computer system for the Array. An autonomous profile shall have exactly one indication configuration service and it shall be hosted on the top level system. It may not be hosted on non-top-level systems (e.g., component computer systems defined in the Multiple Computer System Profile).

- SNIA_IndicationConfigurationCapabilities

There is one instance of SNIA_IndicationConfigurationCapabilities for an SNIA_IndicationConfigurationService. These capabilities define the extrinsic methods supported by the implementation and a set of SupportedFeatures. The possible SupportedSynchronousActions values and their definitions are as follows:

- '2' (None) - None of the IndicationConfigurationService methods are supported.
- '3' (Test Listener) - The TestListener method is supported.

- '4' ("Create and Subscribe) - The CreateAndSubscribe method is supported.
- '5' (Filter Collection Methods) - The methods for managing client defined FilterCollections are supported.

The possible SupportedFeatures values and their definitions are as follows:

- '2' (None) - None of the optional features are supported. Specifically, FilterCollections (either predefined or client defined) are not supported. Filters (either predefined or client defined) are not supported. And semi-fixed Indication Filters are not supported.

Note that in SMI-S, 'none' is only valid for profiles that don't support indications. Any profile that supports indications shall support either or both predefined or client defined indication filters.

- '3' (Predefined Filters) - The implementation has populated a set of predefined IndicationFilters for indications that it supports. These should include those specified by SMI-S, but may include vendor specific IndicationFilters that the implementation supports.
- '4' (Client Defined Filters) - The implementation supports client defined IndicationFilters through CreateInstance (and possibly through the CreateAndSubscribe method). If SupportedFeatures includes '4', but SupportedSynchronousActions does not include '4', it means that only CreateInstance is supported for creation of client defined filters.
- '5' (Predefined Filter Collections) - The implementation has collected its predefined IndicationFilters into FilterCollections. Each predefined FilterCollection corresponds to an SMI-S profile (autonomous or component) and is structured as a two level hierarchy. The top FilterCollection contains the predefined IndicationFilters of the autonomous profile and FilterCollections for the component profiles.
- '6' (Client Defined Filter Collections) - The implementation supports client defined FilterCollections through CreateInstance (and possibly through the CreateAndSubscribe method). If SupportedFeatures includes '6', but SupportedSynchronousActions does not include '4', it means that only CreateInstance is supported for creation of client defined filter collections.
- '7' (Semi-fixed Indication Filters) - The implementation supports semi-fixed IndicationFilters, which means that it can accept client definitions of filters that fit the pattern defined by the semi-fixed IndicationFilter.
- ElementCapabilities

This associates the SNIA_IndicationConfigurationService instance to its SNIA_IndicationConfigurationCapabilities instance.
- ServiceAffectsElement

This associates the IndicationConfigurationService to client defined FilterCollections it manages.

44.2 Fault Management Considerations

EXPERIMENTAL

44.2.1 Indication Correlation

There are cases where many indications are produced in response to a single event. In fact, the indications themselves are correctly viewed as presenting an aspect or view of the event itself and not as a comprehensive representation of the event. AlertIndications provide a means of notification that is more direct to the point than life cycle indications, even though the production of life cycle indications are also important. The subtleties of the effect of the event are better communicated through life cycle indications.

A given event, like a network port communication failure, can itself be reported as an `AlertIndication`. It is also important to communicate the change in status of the port itself through life cycle indications. It is probable that the network port communication failure will cause some function of the device which contains the point to also fail or become degraded. The impact of the failure (or significant state or status change) is of great interest to management clients as it assists in the triage of the error and potentially can also assist HFM aware clients to contain the failure, fence off the failing component, or even prevent a more serious failure of the system in which the component participates, like the failure of business function (like closing the book at quarter end or dropping transactions at Christmas time).

SMI-S provides the mechanism where storage management can be affected without requiring a priori knowledge of the device or application being managed. In this world, the overall system or service component that is most able to assess and report the impact of the failure (or significant state or status change) is the managed device or application itself. Indication correlation provides the mechanism that can be used to asynchronously report the changes brought about by the event.

The mechanism requires that a single indication be the first reporter of the event. This first reporter may be an `AlertIndication` or a life cycle indication. This indication should report the state or status change caused by the event in the simplest and most direct manner. All other indications that report state or status change and are associated directly to the first reporter indications should be correlated to the first reported indication. Indication correlation shall be done by the implementation through reporting the `IndicationIdentifier` of the correlated and previously produced indications in the `CorrelatedIndications` array. The elements in the `CorrelatedIndications` may be in any order. The linkage of indication thusly correlated is like a one-way linked list. The beginning of the correlation link is indicated by the nullness of the `CorrelatedIndications` property.

Indication correlation shall be accomplished in the path of cause and effect or scoping relationships. If indication B is correlated to indication A, then the model change reported by B is caused by or is a side-effect of the model change reported by A. Indication correlation shall not be accomplished by sorting the indications to be correlated by `PerceivedSeverity`. That being said, Indication correlation should not be used to report secondary events, themselves caused by the primary event, and side-effects of the secondary event.

Indication correlation provides important information about the onset of the condition and its immediate impact that may not be retrievable when the client can react. The spread of the effects of the event within a device or application can certainly be faster than maximum speed of the management network.

Indication correlation shall be accomplished through scoping relationships, like the part to group component or dependent to antecedent relationships, or across direct cause and effect relationships for peer components. For example, given that a network port communication failure within a given device causes changes to the status of port, the scoping computer system, the port communications statistics, the status of the network pipe, and the overall communication statistics of the device, then indication correlation shall not report correlation of the network port communication failure to the changes in the overall communications statistics of the device. This requirement is necessary to limit the potentially lengthy correlation and impose undue burden on the implementation without value to the client.

EXPERIMENTAL

44.3 Cascading Considerations

Not Applicable.

44.4 Supported Profiles, Subprofiles and Packages

Related Profiles for Experimental Indication: Not defined in this International Standard.

44.5 Methods of the Profile

44.5.1 Extrinsic Methods of the Profile

44.5.1.1 TestListener Method

The TestListener method allows a client to test that the ListenerDestination is actually reachable from the implementation. It also provides information, in error cases, on why the test failed (e.g., destination not resolvable, Port not reachable, certificate errors, etc.).

```
uint32 IndicationConfigurationService.TestListener(
    [IN] CIM_ListenerDestination REF Desitination );
```

The TestListener method takes as input a reference to a CIM_ListenerDestination class.

The return codes that may be produced by this method are:

- 0 - If a 0 return code is returned, it means the indication was sent. The implementation will send an AlertIndication with the standard message MP22 (Listener Destination Test).
- 1 - The method is not supported (e.g., the provider does not support the function).
- 4 - Failed. If the return code is 4, it means that the provider was not able to send the indication.
- 5 - Invalid parameter. The protocol specified is not a recognized protocol or the destination was not in a valid URI format.

44.5.1.2 CreateAndSubscribe Method

The CreateAndSubscribe method allows a client to create IndicationFilters, ListenerDestinations and Subscriptions in a single extrinsic call. If the instance supplied as input already exist (e.g., an IndicationFilter, FilterCollection and ListenerDestination), then the only element created is the subscription.

```
uint32 IndicationConfigurationService.CreateAndSubscribe(
    [IN, EmbeddedInstance("CIM_IndicationFilter")]
    string IndicationFilter,
    [IN, EmbeddedInstance("CIM_FilterCollection")]
    string FilterCollection,
    [IN, EmbeddedInstance("CIM_ListenerDestination")]
    string ListenerDestination,
    [IN, EmbeddedInstance("CIM_AbstractIndicationSubscription")]
    string SubscriptionData,
    [OUT] CIM_AbstractIndicationSubscription REF Subscription);
```

The CreateAndSubscribe method takes as input a set of 4 embedded Instances provided by the client application:

- CIM_IndicationFilter - This would be filled in if the desired output is an instance of CIM_IndicationSubscription. The method will use properties of the embedded instance to determine if the IndicationFilter exists. If it does not exist and IndicationConfigurationService.SupportedFeatures includes '4', then the method will create the instance.
- CIM_FilterCollection - This would be filled in if the desired output is an instance of CIM_FilterCollectionSubscription. The method will use properties of the embedded instance to determine if the FilterCollection exists. If it does not exist and IndicationConfigurationService.SupportedFeatures includes '6', the method will create the instance (but it will be an empty collection).
- CIM_ListenerDestination - The method will use properties of the embedded instance to determine if the ListenerDestination exists. If it does not exist, the method will create the instance.

- CIM_AbstractIndicationSubscription - As an input embedded instance, the references should be NULL. The other properties of the instance will be used to establish the properties of the subscription created.

The CreateAndSubscribe method output (Subscription) is a reference to the created subscription which can be either an IndicationSubscription or a FilterCollectionSubscription.

The return codes that may be produced by this method are:

- 0 - If a 0 return code is returned, it means the subscription (and any related instances) has been created.
- 1 - Not Supported. The method is not supported (e.g., the provider does not support the function). That is, IndicationConfigurationService.SupportedSynchronousActions does not include '4'.
- 4 - Failed. If the return code is 4, it means that the provider was not able to create the subscription.
- 5 - Invalid Parameter. The implementation does not recognize the value of a parameter.

44.5.1.3 CreateFilterCollection

The CreateFilterCollection method allows a client to create a client specific collection of IndicationFilters (and/or other filter collections) for the purpose of subscribing to all collected indications with one subscription to the FilterCollection.

```
uint32 IndicationConfigurationService.CreateFilterCollection(
    [IN] string FilterCollectionName,
    [IN] CIM_ManagedElement REF Members[],
    [OUT] CIM_FilterCollection REF FilterCollection );
```

The CreateFilterCollection method takes as input the name of the collection and a list of members to be added to the collection:

- FilterCollectionName - FilterCollectionName takes the form 'OrgID:CollectionID' where OrgID is the client vendor and product is part of CollectionID. Predefined collections have an OrgID of 'SNIA', so a client defined filter collection shall not use the prefix 'SNIA'.
- Members[] - This is a list of references to instances of either CIM_IndicationFilter or CIM_FilterCollection. A Member may be an IndicationFilter (a ManagedElement) or another FilterCollection (a Collection).

The CreateFilterCollection method output (FilterCollection) is a reference to the created FilterCollection.

The return codes that may be produced by this method are:

- 0 - If a 0 return code is returned, it means the FilterCollection has been created and the specified members have been added to the collection.
- 1 - The method is not supported (e.g., the provider does not support the function). That is, IndicationConfigurationService.SupportedSynchronousActions does not include '5'.
- 4 - Failed. If the return code is 4, it means that the provider was not able to create the FilterCollection.
- 5 - Invalid parameter. A parameter is not recognized as a valid value. For example, the FilterCollectionName has a prefix of 'SNIA' or one of the Members is not a CIM_IndicationFilter or CIM_FilterCollection.

44.5.1.4 AddFilterToCollection

The AddFilterToCollection method allows a client to add members to an existing client defined FilterCollection.


```
uint32 IndicationConfigurationService.AddFilterToCollection(
    [IN] CIM_ManagedElement REF Members[],
    [IN] CIM_FilterCollection REF FilterCollection );
```

The AddFilterToCollection method takes as input

- Members[] - This is a list of references to instances of either CIM_IndicationFilter or CIM_FilterCollection to be added to the FilterCollection.
- FilterCollection - This is a reference to the (Client Defined) FilterCollection to which the new members are to be added.

The AddFilterToCollection method output is simply success or failure.

The return codes that may be produced by this method are:

- 0 - If a 0 return code is returned, it means the Members have been added to the FilterCollection.
- 1 - The method is not supported (e.g., the provider does not support the function). That is, IndicationConfigurationService.SupportedSynchronousActions does not include '5'.
- 4 - Failed. If the return code is 4, it means that the provider was not able to add the members to the FilterCollection and none of the additions were done.
- 5 - Invalid parameter. A parameter is not recognized as a valid value. For example, one of the Members is not a CIM_IndicationFilter or CIM_FilterCollection.

44.5.1.5 RemoveFilterFromCollection

The RemoveFilterFromCollection method allows a client to remove filters from client defined collections.

```
uint32 IndicationConfigurationService.RemoveFilterFromCollection(
    [IN] CIM_ManagedElement REF Members[],
    [IN] CIM_FilterCollection REF FilterCollection );
```

The RemoveFilterFromCollection method takes as input

- Members[] - This is a list of references to instances of either CIM_IndicationFilter or CIM_FilterCollection to be removed to the FilterCollection.
- FilterCollection - This is a reference to the (Client Defined) FilterCollection to which the members are to be removed.

The RemoveFilterFromCollection method output is simply success or failure.

The return codes that may be produced by this method are:

- 0 - If a 0 return code is returned, it means the Members have been removed to the FilterCollection.
- 1 - The method is not supported (e.g., the provider does not support the function). That is, IndicationConfigurationService.SupportedSynchronousActions does not include '5'.
- 4 - Failed. If the return code is 4, it means that the provider was not able to remove the members to the FilterCollection and none of the removals were done.
- 5 - Invalid parameter. A parameter is not recognized as a valid value. For example, one of the Members is not a CIM_IndicationFilter or CIM_FilterCollection.

44.5.1.6 DeleteFilterCollection

The DeleteFilterCollection method allows a client to remove a client defined FilterCollection.

```
uint32 IndicationConfigurationService.DeleteFilterCollection(
```



```
[IN, Required] CIM_FilterCollection REF FilterCollection.  
[IN] boolean RemoveMembers );
```

The DeleteFilterCollection method takes as input

- FilterCollection - This is a reference to the (Client Defined) FilterCollection to which the members are to be removed.
- RemoveMembers - This boolean, when "true" means the method should imply removal of existing members of the collection. When "false", the method will fail if members exist in the collection.

The DeleteFilterCollection method output (FilterCollection) is simply success or failure.

The return codes that may be produced by this method are:

- 0 - If a 0 return code is returned, it means the FilterCollection has been deleted.
- 1 - The method is not supported (e.g., the provider does not support the function). That is, IndicationConfigurationService.SupportedSynchronousActions does not include '5'.
- 4 - Failed. If the return code is 4, it means that the provider was not able to delete the FilterCollection. For example, this method will return this error if RemoveMembers is 'false' and there are members in the FilterCollection.
- 5 - Invalid parameter. A parameter is not recognized as a valid value. For example, the FilterCollectionName has a prefix of 'SNIA'. That is, this method will return this error if the application attempts to delete a predefined FilterCollection.

44.5.2 Intrinsic Methods of the Profile

44.5.2.1 General

The Experimental Indication Profile is mostly populated by providers and is accessible to clients using basic read and association traversal. However, there are three constructs that may be created by Clients. These are the FilterCollection, MemberOfCollection and the FilterCollectionSubscription. In addition to being able to create them, client may delete them (except "pre-defined" filters which cannot be deleted), and a client may modify any IndicationFilter that was client created.

NOTE The IndicationConfigurationService provides extrinsic methods for creating and managing client defined FilterCollections. The intrinsic methods for creating and maintaining FilterCollections are provided for clients that have a preference to use intrinsic methods.

44.5.2.2 FilterCollection

The CreateInstance and DeleteInstance operations are supported for client defined FilterCollections.

CreateInstance

```
<instanceName>CreateInstance (  
    [IN] <instance> NewInstance  
)
```

On CreateInstance on FilterCollections, the implementation should allow NULL to be specified for key properties (the InstanceID). If key properties are passed in on the CreateInstance, the implementation may ignore the keys.

If the client supplies an CollectionName, the implementation shall persist the property for later use by the client.

If successful, the return value defines the object path of the new CIM Instance relative to the target Namespace (i.e., the Model Path).

If unsuccessful, one of the following status codes shall be returned by this method, where the first applicable error in the list (starting with the first element of the list, and working down) is the error returned. Any additional method-specific interpretation of the error is given in parentheses.

CIM_ERR_ACCESS_DENIED, CIM_ERR_NOT_SUPPORTED,
 CIM_ERR_INVALID_NAMESPACE, CIM_ERR_INVALID_PARAMETER (including missing, duplicate, unrecognized or otherwise incorrect parameters), CIM_ERR_INVALID_CLASS (the CIM Class of which this is to be a new Instance does not exist), CIM_ERR_ALREADY_EXISTS (the CIM Instance already exists), CIM_ERR_FAILED (some other unspecified error occurred).

NOTE If a client attempts to create a FilterCollection that already exists (has the same InstanceID), but other properties are different, then the request will fail. If the Client attempts to create a FilterCollection that has identical properties to an existing FilterCollection instance, it will succeed and CreateInstance need not treat the instance as a separate instance.

DeleteInstance

```
void DeleteInstance (
    [IN] <instanceName> InstanceName
)
```

The InstanceName input parameter defines the name (model path) of the Instance to be deleted.

If successful, the specified FilterCollection Instance shall have been removed.

The deletion of a FilterCollection instance will cause the automatic deletion of any associated MemberOfCollection and FilterCollectionSubscription instances.

If unsuccessful, one of the following status codes shall be returned by this method, where the first applicable error in the list (starting with the first element of the list, and working down) is the error returned. Any additional method-specific interpretation of the error is given in parentheses.

CIM_ERR_ACCESS_DENIED, CIM_ERR_NOT_SUPPORTED (the filter collection is a pre-defined filter collection), CIM_ERR_INVALID_NAMESPACE, CIM_ERR_INVALID_PARAMETER (including missing, duplicate, unrecognized or otherwise incorrect parameters), CIM_ERR_INVALID_CLASS (the CIM Class does not exist in the specified namespace), CIM_ERR_NOT_FOUND (the CIM Class does exist, but the requested CIM Instance does not exist in the specified namespace), CIM_ERR_FAILED (some other unspecified error occurred).

44.5.2.3 MemberOfCollection

The CreateInstance and DeleteInstance operations are supported on MemberOfCollection for associating client defined FilterCollections and their members.

CreateInstance

```
<instanceName>CreateInstance (
    [IN] <instance> NewInstance
)
```

An implementation should populate all fields of references (scheme, hostname, port number, namespace, key) in object path for the instance being created.

The host name portion shall be set to a valid, client-resolvable host name (i.e., DNS) or IPv4 or IPv6 address. Internal names (e.g., /etc/hosts) are not valid. If the host name, then it shall be FQDN (not a short name). This implies the provider shall be configured as a valid DNS client to use host names and cannot rely on an administratively defined name.

If successful, the return value defines the object path of the new CIM Instance relative to the target Namespace (i.e., the Model Path).

Note that for CreateInstance of a MemberOfCollection requires that the FilterCollection instance and the member instances (FilterCollection or IndicationFilter) exist.

If unsuccessful, one of the following status codes shall be returned by this method, where the first applicable error in the list (starting with the first element of the list, and working down) is the error returned. Any additional method-specific interpretation of the error is given in parentheses.

CIM_ERR_ACCESS_DENIED, CIM_ERR_NOT_SUPPORTED (the FilterCollection referenced is a pre-defined filter collection), CIM_ERR_INVALID_NAMESPACE, CIM_ERR_INVALID_PARAMETER (including missing, duplicate, unrecognized or otherwise incorrect parameters), CIM_ERR_INVALID_CLASS (the CIM Class of which this is to be a new Instance does not exist), CIM_ERR_ALREADY_EXISTS (the CIM Instance already exists), CIM_ERR_FAILED (some other unspecified error occurred).

DeleteInstance

```
void DeleteInstance (
    [IN] <instanceName> InstanceName
)
```

The InstanceName input parameter defines the name (model path) of the Instance to be deleted.

If successful, the specified MemberOfCollection Instance shall have been removed.

Deletion of a MemberOfCollection will not cause the deletion of any corresponding FilterCollection or IndicationFilter instances.

If unsuccessful, one of the following status codes shall be returned by this method, where the first applicable error in the list (starting with the first element of the list, and working down) is the error returned. Any additional method-specific interpretation of the error in is given in parentheses.

CIM_ERR_ACCESS_DENIED, CIM_ERR_NOT_SUPPORTED (the FilterCollection referenced is a pre-defined filter collection), CIM_ERR_INVALID_NAMESPACE, CIM_ERR_INVALID_PARAMETER (including missing, duplicate, unrecognized or otherwise incorrect parameters), CIM_ERR_INVALID_CLASS (the CIM Class does not exist in the specified namespace), CIM_ERR_NOT_FOUND (the CIM Class does exist, but the requested CIM Instance does not exist in the specified namespace), CIM_ERR_FAILED (some other unspecified error occurred).

44.5.2.4 FilterCollectionSubscription

The CreateInstance and DeleteInstance operations are supported on FilterCollectionSubscription for associating client defined FilterCollections with ListenerDestinations.

CreateInstance

```
<instanceName>CreateInstance (
    [IN] <instance> NewInstance
)
```

An implementation should populate all fields of references (scheme, hostname, port number, namespace, key) in object path for the instance being created.

The host name portion shall be set to a valid, client-resolvable host name (i.e., DNS) or IPv4 or IPv6 address. Internal names (e.g., /etc/hosts) are not valid. If the host name, then it shall be FQDN (not a short name). This implies the provider shall be configured as a valid DNS client to use host names and cannot rely on an administratively defined name.

If successful, the return value defines the object path of the new CIM Instance relative to the target Namespace (i.e., the Model Path).

Note that for CreateInstance of an FilterCollectionSubscription requires that the ListenerDestinationCIMXML instance and the FilterCollection exist.

If unsuccessful, one of the following status codes shall be returned by this method, where the first applicable error in the list (starting with the first element of the list, and working down) is the error returned. Any additional method-specific interpretation of the error in is given in parentheses.

CIM_ERR_ACCESS_DENIED, CIM_ERR_NOT_SUPPORTED, CIM_ERR_INVALID_NAMESPACE, CIM_ERR_INVALID_PARAMETER (including missing, duplicate, unrecognized or otherwise incorrect parameters), CIM_ERR_INVALID_CLASS (the CIM Class of which this is to be a new Instance does not exist), CIM_ERR_ALREADY_EXISTS (the CIM Instance already exists), CIM_ERR_FAILED (some other unspecified error occurred).

DeleteInstance

```
void DeleteInstance (
    [IN] <instanceName> InstanceName
)
```

The InstanceName input parameter defines the name (model path) of the Instance to be deleted.

If successful, the specified FilterCollectionSubscription instance shall have been removed.

Deletion of a FilterCollectionSubscription will not cause the deletion of any corresponding ListenerDestinationCIMXML or FilterCollection instances.

If unsuccessful, one of the following status codes shall be returned by this method, where the first applicable error in the list (starting with the first element of the list, and working down) is the error returned. Any additional method-specific interpretation of the error in is given in parentheses.

CIM_ERR_ACCESS_DENIED, CIM_ERR_NOT_SUPPORTED, CIM_ERR_INVALID_NAMESPACE, CIM_ERR_INVALID_PARAMETER (including missing, duplicate, unrecognized or otherwise incorrect parameters), CIM_ERR_INVALID_CLASS (the CIM Class does not exist in the specified namespace), CIM_ERR_NOT_FOUND (the CIM Class does exist, but the requested CIM Instance does not exist in the specified namespace), CIM_ERR_FAILED (some other unspecified error occurred).

44.6 Client Considerations and Recipes

44.6.1 General

The use cases in the following subclauses illustrate some of the features (and particularly the methods) of this profile.

44.6.2 Testing a Listener Destination

Table 425 identifies the elements of the use case to test whether a listener destination can receive indications from a CIM Server.

Table 425 - Test that a Listener Destination is Functioning Properly

Use Case Element	Description
Summary	Given an application that listens for indications, test whether or not a CIM_Server can successfully communicate with the application.
Basic Course of Events	1. Start the application that is listening for indications . 2. Tell the CIM_Server to send an indication to the listener application. 3. Verify the CIM_Server sent the indication. 4. Verify that the listening application received the indication.
Alternative Paths	None.
Exception Paths	None
Triggers	Installing a new or upgraded application that listens for indications.
Assumptions	None.
Preconditions	The CIM Server is operational and supports a profile with support for the TestListener function.
Postconditions	The listener application receives the indication and displays it (or logs it).

44.6.3 Discovering predefined IndicationFilters of an implementation

Table 426 identifies the elements of the use case to discover predefined indication filters supported by a profile implementation.

Table 426 - Discovery of Predefined IndicationFilters

Use Case Element	Description
Summary	Given an implementation of an autonomous Profile and its top level ComputerSystem, determine any predefined IndicationFilters it has.
Basic Course of Events	1. Determine if the implementation has IndicationConfigurationCapabilities. 2. If it does, verify that it supports Predefined Indications and/or Predefined FilterCollections. 2a. If Predefined FilterCollections are supported, then look for the Top Level FilterCollection and determine the predefined IndicationFilters supported. 2b. If FilterCollections are not supported (or the IndicationConfigurationService is not supported), then simply enumerate CIM_IndicationFilter in the namespace of the top level ComputerSystem.
Alternative Paths	None.
Exception Paths	None.
Triggers	The administrator (or application) wants to inspect filters that are declared to be supported by an implementation.
Assumptions	None.
Preconditions	The top level system of the profile has been discovered from profile registration and ElementConformsToProfile.
Postconditions	A list of predefined IndicationFilters (possibly by Profile) is produced.

44.6.4 Creating a subscription to a predefined IndicationFilter

Table 427 identifies the elements of the use case to create a subscription to a predefined indication filter.

Table 427 - Create a subscription to a predefined indication filter

Use Case Element	Description
Summary	Given a ListenerDestination and a predefined indication filter, subscribe to the filter.
Basic Course of Events	1. See if the implementation supports the CreateAndSubscribe method. 2. Retrieve the predefined IndicationFilter. 2a. If CreateAndSubscribe is supported, copy properties of the predefined indication filter into an embedded instance and invoke CreateAndSubscribe passing the Destination of the Listener. 2b. If CreateAndSubscribe is not supported, Create the ListenerDestination and Create the IndicationSubscription.
Alternative Paths	None.
Exception Paths	None.
Triggers	Set up a listener to get indications for an indication in the SMI-S Specification.
Assumptions	None.
Preconditions	The top level system of the profile and a listener destination for the application to get the indications.
Postconditions	The subscription is recorded in the CIM Server.

44.6.5 Creating a client defined indication and subscription

Table 428 identifies the elements of the use case to create an indication filter and subscribe to it.

Table 428 - Create an IndicationFilter and subscribe to it

Use Case Element	Description
Summary	Given a top level system of an autonomous profile and a URI for an indication listener create a client defined indication and subscribe to it.
Basic Course of Events	1. Get the IndicationConfigurationService if it exists. 1b. If the service does not exist try to do a CreateInstance for the IndicationFilter (if it succeeds, then continue). 2. If the service exists, get the IndicationConfigurationCapabilities to find out if the implementation supports client defined IndicationFilters. 3. If the capability exists, then do a CreateAndSubscribe for the indication. 3b. Do a CreateInstance on the ListenerDestination and another CreateInstance on the IndicationSubscription.
Alternative Paths	Create a FilterCollection and put the client defined indication filter in that collection.
Exception Paths	None.
Triggers	The administrator wants to listen for a specific indication of his/her choosing.
Assumptions	The implementation supports client defined IndicationFilters.
Preconditions	Use the top level system of the profile and a listener destination for the application to get the indications.
Postconditions	The IndicationFilter is created and a subscription to it is recorded in the CIM Server.

44.6.6 Creating a semi-fixed indication filter

Table 429 identifies the elements of the use case to create a semi-fixed indication filter.

Table 429 - Creation of a semi-fixed Indication filters

Use Case Element	Description
Summary	A client application wants to create an indication filter that has application specific information to include in the filter.
Basic Course of Events	1. Determine if the implementation supports semi-fixed IndicationFilters. 2. Find the semi-fixed IndicationFilter. 2a. Look in the 'SNIA' FilterCollections if they exist. 2b. Enumerate predefined indication filters if not. 3. Do a CreateAndSubscribe for the IndicationFilter substituting the application specific information in the query. 3a. Do a CreateInstance on the filter and the subscription if CreateAndSubscribe is not supported.
Alternative Paths	If none of the new functions are supported try to create the filter using a CreateInstance.
Exception Paths	None.
Triggers	The application wants to restrict the number of indications it receives by adding application specific information to the filter query.
Assumptions	The implementation supports semi-fixed IndicationFilters.
Preconditions	Use the top level system of the profile and a listener destination for the application to get the indications.
Postconditions	The IndicationFilter is created and a subscription to it is recorded in the CIM Server.

44.6.7 Creating a FilterCollection

Table 430 identifies the elements of the use case to create a client defined Filter Collection.

Table 430 - Creation of a client defined FilterCollection

Use Case Element	Description
Summary	Given a top level system of an autonomous profile and a URI for an indication listener and a list of IndicationFilters that the administrator wants to listen to in order to create a client defined FilterCollection and in order to subscribe to it.
Basic Course of Events	1. Determine if the implementation supports client defined FilterCollections. 1a. If not quit, you have no options. 2. Determine if the implementation supports FilterCollection Methods. 2a. If not, go to 3a. 3. Do a CreateFilterCollection to create the FilterCollection. 3a. Do a CreateInstance on the FilterCollection and a bunch of CreateInstances for the MemberOfCollection associations to each of the IndicationFilters. And then do a CreateInstance on FilterCollectionSubscription.
Alternative Paths	None.
Exception Paths	None.
Triggers	The client wants subscriptions to IndicationFilters in a client specific list of filters.
Assumptions	None.
Preconditions	Use the top level system of the profile and a listener destination for the application to get the indications.
Postconditions	The FilterCollection is created and a subscription to it is recorded in the CIM Server.

44.7 Registered Name and Version

Experimental Indication version 1.5.0 (Component Profile).

Specializes SNIA Indication version 1.5.0.

44.8 CIM Elements

44.8.1 General

Table 431 describes the CIM elements for Experimental Indication.

Table 431 - CIM Elements for Experimental Indication

Element Name	Requirement	Description
44.8.2 CIM_AlertIndication	Optional	This is a specialization of the CIM_AlertIndication class in the Indication Profile.
44.8.3 CIM_ElementCapabilities (Indication Config Service to Capabilities)	Mandatory	Experimental. This associates the IndicationConfigurationService to the IndicationConfigurationCapabilities.
44.8.4 CIM_FilterCollection (Client Defined)	Conditional	Experimental. Conditional requirement: Required if SNIA_IndicationConfigurationCapabilities.SupportedFeatures='6' (Client Defined Filter Collections). This is a client defined collection of IndicationFilters to which a client may subscribe.
44.8.5 CIM_FilterCollectionSubscription (Filter Collection Subscription)	Conditional	Experimental. Conditional requirement: Required if SNIA_IndicationConfigurationCapabilities.SupportedFeatures='5' (Predefined Filter Collections) or '6' (Client Defined Filter Collections). This associates the FilterCollection to the system in the referencing profile.
44.8.6 CIM_HostedCollection (Hosted Filter Collection)	Conditional	Experimental. Conditional requirement: Required if SNIA_IndicationConfigurationCapabilities.SupportedFeatures='6' (Client Defined Filter Collections). This associates a client defined FilterCollection to the system in the referencing profile.
44.8.7 CIM_HostedService (Indication Config Service to System)	Mandatory	Experimental. This associates the IndicationConfigurationService to the System in the referencing profile.
44.8.8 CIM_IndicationFilter (client defined)	Optional	This is for 'client defined' CIM_IndicationFilter instances. CIM_IndicationFilter defines the value and format of an indication filter string.
44.8.9 CIM_IndicationFilter (pre-defined)	Optional	This is for 'pre-defined' CIM_IndicationFilter instances. CIM_IndicationFilter defines the value and format of an indication filter string.
44.8.10 CIM_IndicationSubscription	Mandatory	This association is a specialization of the IndicationSubscription as defined in the Indication Profile.
44.8.11 CIM_InstCreation	Optional	This is a specialization of the CIM_InstCreation class in the Indication Profile.

Table 431 - CIM Elements for Experimental Indication

Element Name	Requirement	Description
44.8.12 CIM_InstDeletion	Optional	This is a specialization of the CIM_InstDeletion class in the Indication Profile.
44.8.13 CIM_InstModification	Optional	This is a specialization of the CIM_InstModification class in the Indication Profile.
44.8.14 CIM_ListenerDestinationCIMXML (Indication Handler)	Mandatory	A CIM_ListenerDestinationCIMXML describes the destination for CIM Export Messages to be delivered via CIM-XML. ListenerDestinationCIMXML is subclassed from ListenerDestination.
44.8.15 CIM_ListenerDestinationWSManagement (WS-Man Indication Handler)	Optional	Experimental. A CIM_ListenerDestinationWSManagement describes the destination for CIM Export Messages to be delivered via WS-Man.
44.8.16 CIM_MemberOfCollection (Filter Collection to Filters)	Conditional	Experimental. Conditional requirement: Required if SNIA_IndicationConfigurationCapabilities.SupportedFeatures='6' (Client Defined Filter Collections). This associates a client defined FilterCollection to the Filters in the collection.
44.8.17 SNIA_IndicationConfigurationCapabilities	Mandatory	Experimental. This is the capabilities of the implementation of indications.
44.8.18 SNIA_IndicationConfigurationService	Mandatory	Experimental. This is the indication services of the implementation.
SNIA_IndicationFilterTemplate (semi-fixed).	Conditional	Experimental. Conditional requirement: Required if SNIA_IndicationConfigurationCapabilities.SupportedFeatures='7' (Semi-fixed Indication Filters). This is a template for 'semi-fixed' IndicationFilter instances.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_FilterCollectionSubscription	Optional	Experimental. CQL - This indicates that a subscription to a FilterCollection has been deleted by either explicit user action (DeleteInstance) or by provider clean up.

44.8.2 CIM_AlertIndication

This is a specialization of the CIM_AlertIndication class in the Indication Profile. The class definition specializes the CIM_AlertIndication definition in the Indication profile. Properties or methods not inherited are marked accordingly as '(overridden)' or '(added)' in the left most column.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 432 describes class CIM_AlertIndication.

Table 432 - SMI Referenced Properties/Methods for CIM_AlertIndication

Properties	Flags	Requirement	Description and Notes
IndicationIdentifier (overridden)		Mandatory	An identifier for the Indication used for correlated indications.
CorrelatedIndications		Optional	IndicationIdentifiers whose notifications are correlated with this one.
IndicationTime	N	Mandatory	The time and date of creation of the Indication. The property may be set to NULL if it cannot be determined.
Description		Optional	A free form text description.
AlertingManagedElement (overridden)		Mandatory	The identifying information of the entity for which this Indication is generated. If the element in question is modeled by the profile implementation, then the format for this property should be as a Typed WBEM URI as defined in DSP0207.
AlertingElementFormat		Mandatory	Valid SMI-S values are 0 1 2 ('Unknown' 'Other' 'CIMObjectPath').
AlertType		Mandatory	This shall be 1 2 3 4 5 6 7 8 ('Other' 'Communications Alert' 'Quality of Service Alert' 'Processing Error' 'Device Alert' 'Environmental Alert' 'Model Change' 'Security Alert').
OtherAlertType		Optional	
PerceivedSeverity		Mandatory	This shall be 0 1 2 3 4 5 6 7 ('Unknown', 'Other' 'Information' 'Degraded/Warning' 'Minor' 'Major' 'Critical' 'Fatal/NonRecoverable').
OtherSeverity		Optional	
ProbableCause		Mandatory	Many possible values in a value map. See MOF.
ProbableCauseDescription		Optional	
EventID		Optional	
SystemCreationClassName		Mandatory	
SystemName		Mandatory	The scoping System's Name for the Provider generating this Indication. The SystemName would typically be the name of the system that generates the indication.
ProviderName		Mandatory	
OwningEntity (overridden)	N	Mandatory	A string that uniquely identifies the entity that owns the definition of the format of the Message.
MessageID (overridden)	N	Mandatory	A string that uniquely identifies, within the scope of the OwningEntity, the format of the Message.
Message (overridden)	N	Mandatory	The formatted message (including the MessageArguments).
MessageArguments	N	Optional	An array of strings that contain the dynamic content of the message.
OtherAlertingElementFormat	N	Optional	Not Specified in this version of the Profile.
Trending	N	Optional	Not Specified in this version of the Profile.

Table 432 - SMI Referenced Properties/Methods for CIM_AlertIndication

Properties	Flags	Requirement	Description and Notes
RecommendedActions	N	Optional	Not Specified in this version of the Profile.
EventTime	N	Optional	Not Specified in this version of the Profile.
IndicationFilterName (added)		Mandatory	The list of IndicationFilter.Name values for IndicationFilters or FilterCollections that this indication supports.

44.8.3 CIM_ElementCapabilities (Indication Config Service to Capabilities)

Experimental. This associates the IndicationConfigurationService to the IndicationConfigurationCapabilities.

Requirement: Mandatory

Table 433 describes class CIM_ElementCapabilities (Indication Config Service to Capabilities).

Table 433 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (Indication Config Service to Capabilities)

Properties	Flags	Requirement	Description and Notes
Capabilities		Mandatory	The indication capabilities instance associated with the indication configuration service.
ManagedElement		Mandatory	The indication configuration service.

44.8.4 CIM_FilterCollection (Client Defined)

Experimental. This is a client defined collection of IndicationFilters to which a client may subscribe. An implementation would indicate support for client defined FilterCollections by the SNIA_IndicationConfigurationCapabilities.FeaturesSupported = '6' (Client Defined Filter Collections).

Requirement: Required if SNIA_IndicationConfigurationCapabilities.SupportedFeatures='6' (Client Defined Filter Collections).

Table 434 describes class CIM_FilterCollection (Client Defined).

Table 434 - SMI Referenced Properties/Methods for CIM_FilterCollection (Client Defined)

Properties	Flags	Requirement	Description and Notes
InstanceID		Mandatory	Shall specify the unique identifier for an instance of this class within the Implementation namespace.
CollectionName		Mandatory	The value of CollectionName shall be constructed using the following algorithm: OrgID:CollectionID where OrgID and CollectionID are separated by a colon ':'.

44.8.5 CIM_FilterCollectionSubscription (Filter Collection Subscription)

Experimental. This associates the FilterCollection to the system in the referencing profile.

Requirement: Required if SNIA_IndicationConfigurationCapabilities.SupportedFeatures='5' (Predefined Filter Collections) or '6' (Client Defined Filter Collections).

Table 435 describes class CIM_FilterCollectionSubscription (Filter Collection Subscription).

Table 435 - SMI Referenced Properties/Methods for CIM_FilterCollectionSubscription (Filter Collection Subscription)

Properties	Flags	Requirement	Description and Notes
OnFatalErrorPolicy		Mandatory	
OtherOnFatalErrorPolicy		Optional	This should contain a description of the fatal error policy if OnFatalErrorPolicy=1 (Other).
FailureTriggerTimeInterval		Mandatory	Specifies minimum delay before OnFatalErrorPolicy is implemented.
SubscriptionState		Mandatory	
OtherSubscriptionState		Optional	This should contain a description of the subscription state if SubscriptionState=1 (Other).
RepeatNotificationPolicy		Mandatory	This shall be 2 (None), 3 (Suppress), or 4 (Delay).
RepeatNotificationInterval		Optional	This should be provided if the value of RepeatNotificationPolicy is 3 (Suppress) or 4 (Delay).
RepeatNotificationGap		Optional	This should be provided if the value of RepeatNotificationPolicy is 4 (Delay).
RepeatNotificationCount		Optional	This should be provided if the value of RepeatNotificationPolicy is 3 (Suppress) or 4 (Delay).
Filter		Mandatory	Reference to the FilterCollection.
Handler		Mandatory	Reference to the ListenerDestination.

44.8.6 CIM_HostedCollection (Hosted Filter Collection)

Experimental. This associates a client defined FilterCollection to the system in the referencing profile.

Requirement: Required if SNIA_IndicationConfigurationCapabilities.SupportedFeatures='6' (Client Defined Filter Collections).

Table 436 describes class CIM_HostedCollection (Hosted Filter Collection).

Table 436 - SMI Referenced Properties/Methods for CIM_HostedCollection (Hosted Filter Collection)

Properties	Flags	Requirement	Description and Notes
Dependent		Mandatory	Reference to the FilterCollection.
Antecedent		Mandatory	Reference to the 'Top level' System that hosts the collection.

44.8.7 CIM_HostedService (Indication Config Service to System)

Experimental. This associates the IndicationConfigurationService to the System in the referencing profile.

Requirement: Mandatory.

Table 437 describes class CIM_HostedService (Indication Config Service to System).

**Table 437 - SMI Referenced Properties/Methods for CIM_HostedService
(Indication Config Service to System)**

Properties	Flags	Requirement	Description and Notes
Antecedent		Mandatory	The hosting system.
Dependent		Mandatory	The Indication configuration service hosted on the system.

44.8.8 CIM_IndicationFilter (client defined)

CIM_IndicationFilter instances that are 'client defined' are IndicationFilters that are created by a client using CreateInstance. If a profile implementation can support client defined IndicationFilters, the implementation would support 'client defined' IndicationFilter instances. The implementation shall support 'client defined' filters that are defined by SMI-S profile as mandatory, but may also support additional filters supported by the implementation.

CIM_IndicationFilter is subclassed from CIM_ManagedElement.

Created By: CreateInstance

Modified By: ModifyInstance

Deleted By: DeleteInstance

Requirement: Optional.

Table 438 describes class CIM_IndicationFilter (client defined).

Table 438 - SMI Referenced Properties/Methods for CIM_IndicationFilter (client defined)

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
Name		Mandatory	This should take the form OrgID ":" RegisteredName ":" UniqueID. For more details, see 43.1.2 IndicationFilter Names.
SourceNamespace	N	Optional	Deprecated. For instances in the InteropNamespace, this shall be the namespace where the indications are to originate. For instances in the implementation namespace where the indications are to originate (e.g., the namespace of the profile that supports the filter), this may be NULL to indicate the Filter is registered in the Namespace where the indications originate.
SourceNamespaces	N	Mandatory	This should be all the namespaces where the indications may originate.
Query		Mandatory	A string that specifies (in QueryLanguage terms) which indications are to be delivered to the ListenerDestinations.
QueryLanguage		Mandatory	This shall be 'DMTF:CQL' for CQL queries, but may be 'WQL' or 'SMI-S V1.0'. WQL and SMI-S V1.0 are deprecated in favor of 'DMTF:CQL'.

Table 438 - SMI Referenced Properties/Methods for CIM_IndicationFilter (client defined)

Properties	Flags	Requirement	Description and Notes
ElementName		Optional	A Client Defined user friendly string that identifies the Indication Filter.
Caption	N	Optional	Not Specified in this version of the Profile.
Description	N	Optional	Not Specified in this version of the Profile.

44.8.9 CIM_IndicationFilter (pre-defined)

CIM_IndicationFilter instances that are 'pre-defined' are IndicationFilters that are populated automatically by the profile provider. If a profile implementation cannot support client defined IndicationFilters, the implementation can populate its model with 'pre-defined' IndicationFilter instances. 'Pre-defined' filters shall include those that are required by the profile, but may also contain additional filters supported by the implementation.

CIM_IndicationFilter is subclassed from CIM_ManagedElement.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional.

Table 439 describes class CIM_IndicationFilter (pre-defined).

Table 439 - SMI Referenced Properties/Methods for CIM_IndicationFilter (pre-defined)

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
Name		Mandatory	This should take the form OrgID ":" RegisteredName ":" UniqueID. For more details, see 43.1.2 IndicationFilter Names.
SourceNamespace	N	Optional	Deprecated. For instances in the InteropNamespace, this shall be the namespace where the indications are to originate. For instances in the implementation namespace where the indications are to originate (e.g., the namespace of the profile that supports the filter), this may be NULL to indicate the Filter is registered in the Namespace where the indications originate.
SourceNamespaces	N	Mandatory	This should be all the namespaces where the indications may originate.
Query		Mandatory	A string that specifies (in QueryLanguage terms) which indications are to be delivered to the ListenerDestinations.
QueryLanguage		Mandatory	This shall be 'DMTF:CQL' for CQL queries, but may be 'WQL' or 'SMI-S V1.0'. WQL and SMI-S V1.0 are deprecated in favor of 'DMTF:CQL'.
ElementName	N	Optional	SMI-S does not specify this property for pre-defined IndicationFilters.
Caption	N	Optional	Not Specified in this version of the Profile.
Description	N	Optional	Not Specified in this version of the Profile.