

BS ISO/IEC 24775-7:2014



BSI Standards Publication

Information technology — Storage management

Part 7: Host elements

bsi.

...making excellence a habit.
AVA Documents™
Document Sharing Hub

National foreword

This British Standard is the UK implementation of ISO/IEC 24775-7:2014.

The UK participation in its preparation was entrusted to Technical Committee ICT/-/1, Information systems co-ordination.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

© The British Standards Institution 2014.
Published by BSI Standards Limited 2014

ISBN 978 0 580 84214 6
ICS 35.200

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 31 December 2014.

Amendments/corrigenda issued since publication

Date	Text affected
------	---------------



INTERNATIONAL STANDARD



**Information technology – Storage management –
Part 7: Host elements**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

PRICE CODE

H

ICS 35.200

ISBN 978-2-8322-1955-3

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

Contents

FOREWORD.....	13
Introduction	15
1 Scope	17
2 Normative References.....	18
3 Terms and definitions	18
4 Typographical conventions.....	20
4.1 Maturity model.....	20
4.2 Experimental maturity level	20
4.3 Implemented maturity level	20
4.4 Stable maturity level.....	21
4.5 Finalized maturity level.....	21
4.6 Deprecated material	21
5 Disk Partition Subprofile	23
5.1 Description	23
5.1.1 General.....	23
5.1.2 Background on X86 MBR Partitions	24
5.2 Health and Fault Management Considerations	28
5.3 Supported Subprofiles and Packages.....	29
5.4 Methods of the Profile	29
5.4.1 SetPartitionStyle	29
5.4.2 CreateOrModifyPartition	29
5.5 Client Considerations and Recipes	30
5.5.1 General.....	30
5.5.2 Create New Partition Using All Available Space at End of Volume.....	30
5.6 Registered Name and Version	33
5.7 CIM Elements.....	33
5.7.1 General.....	33
5.7.2 CIM_BasedOn (Partition to Extent)	34
5.7.3 CIM_BasedOn (Partition to Partition)	34
5.7.4 CIM_DiskPartitionConfigurationCapabilities.....	34
5.7.5 CIM_DiskPartitionConfigurationService	35
5.7.6 CIM_ElementCapabilities	35
5.7.7 CIM_GenericDiskPartition	35
5.7.8 CIM_HostedService	36
5.7.9 CIM_InstalledPartitionTable (Capabilities to Extent)	36
5.7.10 CIM_InstalledPartitionTable (Capabilities to Partition)	36
5.7.11 CIM_LogicalDisk	37
5.7.12 CIM_LogicalDiskBasedOnPartition	37
5.7.13 CIM_StorageExtent	37
5.7.14 CIM_SystemDevice (System to Extent)	38
5.7.15 CIM_SystemDevice (System to LogicalDisk)	38
5.7.16 CIM_SystemDevice (System to Partition)	38
6 FC HBA Profile	40
6.1 Synopsis.....	40
6.2 Description	40
6.3 Implementation.....	40
6.3.1 General.....	40
6.3.2 Modeling SCSI Protocol Support.....	41
6.3.3 Persistent Binding	42
6.3.4 LED Blink.....	43
6.4 Health and Fault Management.....	43
6.5 Methods	44

6.5.1	Extrinsic Methods of this Profile	44
6.5.2	StorageNameBindingService.CreateStorageNameBinding	44
6.5.3	StorageNameBindingService.CreateOSStorageNameBinding	44
6.5.4	CIM_AlarmDevice.SetAlarmState	45
6.5.5	Intrinsic Methods of this Profile	45
6.6	Client Considerations and Recipes	46
6.6.1	General.....	46
6.6.2	Discovery HBA Topology and Attributes	46
6.6.3	Get the statistics for each port.....	48
6.6.4	Define a persistent binding to a target PWWN	49
6.6.5	Define a persistent binding to an LUID.....	51
6.6.6	Blink the LED.....	52
6.7	CIM Elements.....	53
6.7.1	General.....	53
6.7.2	CIM_AlarmDevice	54
6.7.3	CIM_AssociatedAlarm	55
6.7.4	CIM_ComputerSystem	55
6.7.5	CIM_ControlledBy	56
6.7.6	CIM_ElementCapabilities (Capabilities to FCPort).....	56
6.7.7	CIM_ElementCapabilities (Capabilities to System)	56
6.7.8	CIM_ElementSettingData	57
6.7.9	CIM_ElementSoftwareIdentity (Driver).....	57
6.7.10	CIM_ElementSoftwareIdentity (FCode/BIOS)	57
6.7.11	CIM_ElementSoftwareIdentity (Firmware)	58
6.7.12	CIM_FCPort	58
6.7.13	CIM_HostedCollection.....	58
6.7.14	CIM_HostedService	59
6.7.15	CIM_InstalledSoftwareIdentity (Driver).....	59
6.7.16	CIM_LogicalPortGroup.....	59
6.7.17	CIM_MemberOfCollection	60
6.7.18	CIM_OSStorageNameBinding	60
6.7.19	CIM_PhysicalPackage	61
6.7.20	CIM_PortController.....	61
6.7.21	CIM_Product	61
6.7.22	CIM_ProductPhysicalComponent	62
6.7.23	CIM_Realizes	62
6.7.24	CIM_ServiceAvailableToElement.....	62
6.7.25	CIM_SoftwareIdentity (Driver)	63
6.7.26	CIM_SoftwareIdentity (FCode/BIOS)	63
6.7.27	CIM_SoftwareIdentity (Firmware).....	63
6.7.28	CIM_StorageNameBinding.....	64
6.7.29	CIM_StorageNameBindingCapabilities	65
6.7.30	CIM_StorageNameBindingService.....	65
6.7.31	CIM_SystemDevice (Associates System to PortController).....	65
7	Storage HBA Profile	67
7.1	Synopsis.....	67
7.2	Description	67
7.3	Implementation.....	67
7.3.1	Health and Fault Management Consideration	67
7.3.2	Cascading Considerations	67
7.3.3	Storage HBA Model Overview.....	67
7.3.4	CIM_ComputerSystem	68
7.3.5	Profile Registration Profile	68
7.3.6	Generic Initiator Ports Profile	69

7.3.7	Software Inventory Profile	69
7.3.8	Software Update Profile	71
7.3.9	HBA Hot Swap Events	71
7.3.10	Physical Asset Profile	71
7.3.11	Modeling Attached Disk, Tape, and Optical Drives	72
7.3.12	Virtual System Considerations	72
7.3.13	Fibre Channel HBAs	72
7.4	Methods of the Profile	73
7.4.1	General	73
7.4.2	Profile Conventions for Operations	73
7.5	Use Cases	73
7.6	CIM Elements	74
7.6.1	General	74
7.6.2	CIM_ControlledBy	74
7.6.3	CIM_PortController	75
7.6.4	CIM_Product	75
7.6.5	CIM_ProductElementComponent	75
7.6.6	CIM_Realizes	76
7.6.7	CIM_SystemDevice	76
8	Host Discovered Resources Profile	77
8.1	Description	77
8.1.1	General	77
8.1.2	Host Disk Extent Class Name Conventions	78
8.1.3	Discovered Hardware Resources	78
8.2	Health and Fault Management Considerations	82
8.3	Cascading Considerations	82
8.4	Supported Subprofiles and Packages	82
8.5	Extrinsic Methods of the Profile – StorageConfigurationService.ScsiScan	82
8.6	Client Considerations and Recipes	82
8.7	Registered Name and Version	84
8.8	CIM Elements	84
8.8.1	General	84
8.8.2	CIM_ATAInitiatorTargetLogicalUnitPath	85
8.8.3	CIM_ATAProtocolEndpoint	85
8.8.4	CIM_ComputerSystem	86
8.8.5	CIM_HostedAccessPoint	86
8.8.6	CIM_LogicalDisk	87
8.8.7	CIM_SCSIArbitraryLogicalUnit	87
8.8.8	CIM_SCSIInitiatorTargetLogicalUnitPath	88
8.8.9	CIM_SCSIProtocolEndpoint	88
8.8.10	CIM_StorageExtent	88
8.8.11	CIM_SystemDevice	89
8.8.12	CIM_TapeDrive	89
8.8.13	SNIA_SBInitiatorTargetLogicalUnitPath	90
8.8.14	SNIA_SBProtocolEndpoint	90
9	Host Hardware RAID Controller Profile	92
9.1	Synopsis	92
9.2	Description	92
9.3	Implementation	93
9.3.1	Relationship to autonomous profiles	93
9.3.2	CIM_PortController	93
9.3.3	CIM_ComputerSystem	93
9.3.4	CIM_AlarmDevice	94
9.3.5	Server Profile	94

9.3.6	Profile Registration	94
9.3.7	Profile Discovery and Advertisement	95
9.3.8	Physical Asset Profile	95
9.3.9	Storage Enclosure Profile	95
9.3.10	Implementation of Block Services Package	95
9.3.11	Implementation of DAPort and SCSIProtocolController	97
9.3.12	Implementation of Software Inventory Profile	98
9.3.13	Implementation of Initiator Ports Profiles	99
9.3.14	Models for Imported Storage	100
9.3.15	Implementation of Extent Composition Profile	102
9.3.16	Disk Sparing	102
9.3.17	Multi-function controllers	102
9.3.18	Health and Fault Management Consideration	103
9.3.19	Cascading Considerations	104
9.4	Methods	104
9.4.1	Extrinsic Methods of the Profile	104
9.4.2	Intrinsic Methods of this Profile	104
9.5	Use Cases	104
9.6	CIM Elements	105
9.6.1	General	105
9.6.2	CIM_AlarmDevice	106
9.6.3	CIM_AssociatedAlarm	107
9.6.4	CIM_ComputerSystem (Host Hardware RAID Controller)	107
9.6.5	CIM_ComputerSystemPackage (Associates PhysicalPackage to ComputerSystem)	108
9.6.6	CIM_ControlledBy	108
9.6.7	CIM_LogicalIdentity	108
9.6.8	CIM_MediaAccessDevice	109
9.6.9	CIM_PortController	109
9.6.10	CIM_Product	109
9.6.11	CIM_ProductPhysicalComponent	110
9.6.12	CIM_ProtocolControllerForUnit (Extent or MediaAccessDevice)	110
9.6.13	CIM_ProtocolControllerForUnit (Volume)	111
9.6.14	CIM_Realizes (Associates PhysicalPackage to PortController)	111
9.6.15	CIM_SAPAvailableForElement	111
9.6.16	CIM_SCSIProtocolController	112
9.6.17	CIM_SoftwareIdentity (Driver)	112
9.6.18	CIM_SoftwareIdentity (FCode/BIOS)	112
9.6.19	CIM_SoftwareIdentity (Firmware)	113
9.6.20	CIM_StorageExtent	113
9.6.21	CIM_SystemComponent	114
9.6.22	CIM_SystemDevice (Associates System to AlarmDevice)	114
9.6.23	CIM_SystemDevice (Associates controller system to PortController)	114
9.6.24	CIM_SystemDevice (System to SCSIProtocolController)	115
10	iSCSI Initiator Profile	116
10.1	Description	116
10.1.1	General	116
10.1.2	Sessions and Connections	117
10.1.3	Durable Names and Correlatable IDs of the Profile	118
10.2	Health and Fault Management Considerations	119
10.3	Supported Subprofiles and Packages	119
10.4	Methods of the Profile	119
10.5	Client Considerations and Recipes	119
10.5.1	Add an additional NIC port	119
10.5.2	Find the health of an initiator	119

10.5.3	Enable/disable header and data digest	119
10.6	Registered Name and Version	119
10.7	CIM Elements.....	120
10.7.1	General.....	120
10.7.2	CIM_BindsTo (TCPProtocolEndpoint to IPProtocolEndpoint)	121
10.7.3	CIM_BindsTo (iSCSIProtocolEndpoint to TCPProtocolEndpoint)	121
10.7.4	CIM_ComputerSystem (Host Hardware RAID Controller).....	121
10.7.5	CIM_ControlledBy (Host Hardware RAID Controller).....	122
10.7.6	CIM_DeviceSAPImplementation (EthernetPort to IPProtocolEndpoint).....	122
10.7.7	CIM_DeviceSAPImplementation (EthernetPort to iSCSIProtocolEndpoint)	123
10.7.8	CIM_ElementSoftwareIdentity (Host Hardware RAID Controller)	123
10.7.9	CIM_EndpointOfNetworkPipe (Between iSCSI Session and iSCSIProtocolEndpoint)	123
10.7.10	CIM_EndpointOfNetworkPipe (Between iSCSI connection and TCPProtocolEndpoint).....	124
10.7.11	CIM_InstalledSoftwareIdentity (Host Hardware RAID Controller)	124
10.7.12	CIM_NetworkPipeComposition (Host Hardware RAID Controller).....	124
10.7.13	CIM_PhysicalPackage (Host Hardware RAID Controller).....	124
10.7.14	CIM_PortController (Host Hardware RAID Controller)	125
10.7.15	CIM_Product (Host Hardware RAID Controller).....	125
10.7.16	CIM_ProductPhysicalComponent (Host Hardware RAID Controller).....	125
10.7.17	CIM_ProtocolControllerForPort (Host Hardware RAID Controller)	126
10.7.18	CIM_Realizes (Host Hardware RAID Controller)	126
10.7.19	CIM_SAPAvailableForElement (Host Hardware RAID Controller).....	126
10.7.20	CIM_SCSIProtocolController (Host Hardware RAID Controller)	127
10.7.21	CIM_SoftwareIdentity (Host Hardware RAID Controller)	127
10.7.22	CIM_SystemDevice (to EthernetPort)	128
10.7.23	CIM_SystemDevice (to PortController)	128
10.7.24	CIM_SystemDevice (to ProtocolController)	128
10.7.25	CIM_iSCSIConnection (Host Hardware RAID Controller).....	128
10.7.26	CIM_iSCSISession (Host Hardware RAID Controller)	129
11	SCSI Multipath Management Subprofile	131
11.1	Description	131
11.1.1	General.....	131
11.1.2	Asymmetric Multipath Target Devices	132
11.2	Health and Fault Management Considerations.....	133
11.3	Cascading Considerations	133
11.4	Supported Subprofiles and Packages.....	133
11.5	Methods of the Profile	133
11.5.1	General.....	133
11.5.2	SCSIPathConfigurationService.SetTPGAccess	133
11.5.3	SCSIPathConfigurationService.SetLoadBalanceAlgorithm.....	133
11.5.4	SCSIPathConfigurationService.AssignLogicalUnitToPortGroup	133
11.5.5	SCSIPathConfigurationService.SetOverridePath.....	134
11.5.6	SCSIPathConfigurationService.CancelOverridePath	134
11.6	Client Considerations and Recipes	134
11.6.1	Discover All Paths to a Disk Volume	134
11.6.2	Force Failover or change Load Balancing on a volume	136
11.6.3	Change a LogicalDisk's LoadBalance Algorithm.....	138
11.7	Registered Name and Version	140
11.8	CIM Elements.....	140
11.8.1	General.....	140
11.8.2	CIM_ConcreteComponent.....	141
11.8.3	CIM_ConcreteDependency	141
11.8.4	CIM_ElementCapabilities	141

11.8.5	CIM_ElementSettingData	142
11.8.6	CIM_ElementSoftwareIdentity (Driver)	142
11.8.7	CIM_ElementSoftwareIdentity (MP API Plugin)	142
11.8.8	CIM_HostedService	143
11.8.9	CIM_MemberOfCollection	143
11.8.10	CIM_Product	143
11.8.11	CIM_SCSIMultipathConfigurationCapabilities	144
11.8.12	CIM_SCSIMultipathSettings	144
11.8.13	CIM_SCSIPathConfigurationService	145
11.8.14	CIM_SCSITargetPortGroup	145
11.8.15	CIM_ServiceAvailableToElement	146
11.8.16	CIM_SoftwareIdentity (Driver)	146
11.8.17	CIM_SoftwareIdentity (MP API Plugin)	147
12	SB Multipath Management Profile	148
12.1	General remark	148
12.2	Description	148
12.3	Health and Fault Management Consideration	149
12.4	Cascading Considerations	149
12.5	Supported Profiles, Subprofiles, and Packages	149
12.6	Methods	149
12.6.1	Extrinsic Methods of this Profile	149
12.6.2	Intrinsic Methods of this Profile	149
12.7	Client Considerations and Recipes	149
12.8	Registered Name and Version	150
12.9	CIM Elements	150
12.9.1	General	150
12.9.2	SNIA_SBInitiatorTargetLogicalUnitPath	150
Annex A	(informative) SMI-S Information Model	151
Bibliography		153

Figure 1 - Experimental maturity level tag	20
Figure 2 - Implemented maturity level tag.....	21
Figure 3 - Stable maturity level tag	21
Figure 4 - Deprecated tag	22
Figure 5 - Disk Partition Class Hierarchy	24
Figure 6 - Disk Partition Class Diagram	24
Figure 7 - Disk MBR Partition Example	25
Figure 8 - MBR Partition Instance Diagram	26
Figure 9 - MBR and VTOC Partition Instance Diagram	27
Figure 10 - Partition Instance Diagram for Size/Address Rules.....	28
Figure 11 - FC HBA Instance Diagram	41
Figure 12 - HBA Card with Two Ports.....	42
Figure 13 - Persistent Binding Model.....	43
Figure 14 - Model Overview.....	68
Figure 15 - Profile Registration Profile	69
Figure 16 - Software Inventory Profile in Storage HBA.....	71
Figure 17 - HBA Card with Physical Classes.....	72
Figure 18 - Host Discovered Resources Block Diagram.....	77
Figure 19 - Host Discovered Resources Class Diagram	79
Figure 20 - Single SPI Disk Model	80
Figure 21 - Three FCP Logical Unit Instance Diagram	80
Figure 22 - ATA Discovered Resource Model	81
Figure 23 - SB Host Discovered Resources	81
Figure 24 - Host Hardware RAID Controller Package Diagram.....	93
Figure 25 - Alarms in Host Hardware RAID Controller	94
Figure 26 - Profile Registration with Host Hardware RAID Controller and Base Server Profiles	94
Figure 27 - Implementation of Physical Asset Profile	95
Figure 28 - Block Services Package in Host Hardware RAID Controller	96
Figure 29 - DAPort Subprofile in Host Hardware Controller	98
Figure 30 - Software Inventory Profile in Host Hardware RAID Controller	98
Figure 31 - Initiator Port profiles and Disk Drive Lite Subprofile	99
Figure 32 - Model for Imported Disks.....	100
Figure 33 - Imported Virtual Volumes	101
Figure 34 - Device "Pass Through" Example.....	102
Figure 35 - Example of Mutli-Function Controllers.....	103
Figure 36 - iSCSI Product and Package Model	117
Figure 37 - iSCSI Sessions and Connections Model	118
Figure 38 - iSCSI Initiator Node	118
Figure 39 - Multipath Management Class Diagram	131
Figure 40 - Four Path Instance Diagram.....	132
Figure 41 - Four SB Channel Instance Diagram.....	148
Figure 42 - Two SB Channel, Three Volume Instance Diagram.....	149

Table 1 - Capabilities Properties	27
Table 2 - CIM Elements for Disk Partition	33
Table 3 - SMI Referenced Properties/Methods for CIM_BasedOn (Partition to Extent)	34
Table 4 - SMI Referenced Properties/Methods for CIM_BasedOn (Partition to Partition)	34
Table 5 - SMI Referenced Properties/Methods for CIM_DiskPartitionConfigurationCapabilities	35
Table 6 - SMI Referenced Properties/Methods for CIM_DiskPartitionConfigurationService	35
Table 7 - SMI Referenced Properties/Methods for CIM_ElementCapabilities	35
Table 8 - SMI Referenced Properties/Methods for CIM_GenericDiskPartition	36
Table 9 - SMI Referenced Properties/Methods for CIM_HostedService	36
Table 10 - SMI Referenced Properties/Methods for CIM_InstalledPartitionTable (Capabilities to Extent)	36
Table 11 - SMI Referenced Properties/Methods for CIM_InstalledPartitionTable (Capabilities to Partition)	37
Table 12 - SMI Referenced Properties/Methods for CIM_LogicalDisk	37
Table 13 - SMI Referenced Properties/Methods for CIM_LogicalDiskBasedOnPartition	37
Table 14 - SMI Referenced Properties/Methods for CIM_StorageExtent	38
Table 15 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to Extent)	38
Table 16 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to LogicalDisk)	38
Table 17 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to Partition)	39
Table 18 - Related Profiles for FC HBA	40
Table 19 - CIM Elements for FC HBA	53
Table 20 - SMI Referenced Properties/Methods for CIM_AlarmDevice	55
Table 21 - SMI Referenced Properties/Methods for CIM_AssociatedAlarm	55
Table 22 - SMI Referenced Properties/Methods for CIM_ComputerSystem	55
Table 23 - SMI Referenced Properties/Methods for CIM_ControlledBy	56
Table 24 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (Capabilities to FCPort)	56
Table 25 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (Capabilities to System)	57
Table 26 - SMI Referenced Properties/Methods for CIM_ElementSettingData	57
Table 27 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity (Driver)	57
Table 28 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity (FCode/BIOS)	58
Table 29 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity (Firmware)	58
Table 30 - SMI Referenced Properties/Methods for CIM_FCPort	58
Table 31 - SMI Referenced Properties/Methods for CIM_HostedCollection	58
Table 32 - SMI Referenced Properties/Methods for CIM_HostedService	59
Table 33 - SMI Referenced Properties/Methods for CIM_InstalledSoftwareIdentity (Driver)	59
Table 34 - SMI Referenced Properties/Methods for CIM_LogicalPortGroup	59
Table 35 - SMI Referenced Properties/Methods for CIM_MemberOfCollection	60
Table 36 - SMI Referenced Properties/Methods for CIM_OSStorageNameBinding	60
Table 37 - SMI Referenced Properties/Methods for CIM_PhysicalPackage	61
Table 38 - SMI Referenced Properties/Methods for CIM_PortController	61
Table 39 - SMI Referenced Properties/Methods for CIM_Product	62
Table 40 - SMI Referenced Properties/Methods for CIM_ProductPhysicalComponent	62
Table 41 - SMI Referenced Properties/Methods for CIM_Realizes	62
Table 42 - SMI Referenced Properties/Methods for CIM_ServiceAvailableToElement	63
Table 43 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity (Driver)	63
Table 44 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity (FCode/BIOS)	63
Table 45 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity (Firmware)	64
Table 46 - SMI Referenced Properties/Methods for CIM_StorageNameBinding	64
Table 47 - SMI Referenced Properties/Methods for CIM_StorageNameBindingCapabilities	65
Table 48 - SMI Referenced Properties/Methods for CIM_StorageNameBindingService	65
Table 49 - SMI Referenced Properties/Methods for CIM_SystemDevice (Associates System to PortController)	66

Table 50 - Related Profiles for Storage HBA	67
Table 51 - CIM_PortController	73
Table 52 - CIM_SystemDevice	73
Table 53 - CIM Elements for Storage HBA	74
Table 54 - SMI Referenced Properties/Methods for CIM_ControlledBy	75
Table 55 - SMI Referenced Properties/Methods for CIM_PortController	75
Table 56 - SMI Referenced Properties/Methods for CIM_Product	75
Table 57 - SMI Referenced Properties/Methods for CIM_ProductElementComponent	76
Table 58 - SMI Referenced Properties/Methods for CIM_Realizes	76
Table 59 - SMI Referenced Properties/Methods for CIM_SystemDevice	76
Table 60 - Supported Profiles for Host Discovered Resources	82
Table 61 - CIM Elements for Host Discovered Resources	84
Table 62 - SMI Referenced Properties/Methods for CIM_ATAInitiatorTargetLogicalUnitPath	85
Table 63 - SMI Referenced Properties/Methods for CIM_ATAProtocolEndpoint	86
Table 64 - SMI Referenced Properties/Methods for CIM_ComputerSystem	86
Table 65 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint	86
Table 66 - SMI Referenced Properties/Methods for CIM_LogicalDisk	87
Table 67 - SMI Referenced Properties/Methods for CIM_SCSIArbitraryLogicalUnit	87
Table 68 - SMI Referenced Properties/Methods for CIM_SCSIInitiatorTargetLogicalUnitPath	88
Table 69 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint	88
Table 70 - SMI Referenced Properties/Methods for CIM_StorageExtent	89
Table 71 - SMI Referenced Properties/Methods for CIM_SystemDevice	89
Table 72 - SMI Referenced Properties/Methods for CIM_TapeDrive	89
Table 73 - SMI Referenced Properties/Methods for SNIA_SBInitiatorTargetLogicalUnitPath	90
Table 74 - SMI Referenced Properties/Methods for SNIA_SBProtocolEndpoint	91
Table 75 - Related Profiles for Host Hardware RAID Controller	92
Table 76 - CIM Elements for Host Hardware RAID Controller	105
Table 77 - SMI Referenced Properties/Methods for CIM_AlarmDevice	106
Table 78 - SMI Referenced Properties/Methods for CIM_AssociatedAlarm	107
Table 79 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Host Hardware RAID Controller)	107
Table 80 - SMI Referenced Properties/Methods for CIM_ComputerSystemPackage (Associates PhysicalPackage to ComputerSystem)	108
Table 81 - SMI Referenced Properties/Methods for CIM_ControlledBy	108
Table 82 - SMI Referenced Properties/Methods for CIM_LogicalIdentity	108
Table 83 - SMI Referenced Properties/Methods for CIM_MediaAccessDevice	109
Table 84 - SMI Referenced Properties/Methods for CIM_PortController	109
Table 85 - SMI Referenced Properties/Methods for CIM_Product	110
Table 86 - SMI Referenced Properties/Methods for CIM_ProductPhysicalComponent	110
Table 87 - SMI Referenced Properties/Methods for CIM_ProtocolControllerForUnit (Extent or MediaAccessDevice)	111
Table 88 - SMI Referenced Properties/Methods for CIM_ProtocolControllerForUnit (Volume)	111
Table 89 - SMI Referenced Properties/Methods for CIM_Realizes (Associates PhysicalPackage to PortController)	111
Table 90 - SMI Referenced Properties/Methods for CIM_SAPAvailableForElement	112
Table 91 - SMI Referenced Properties/Methods for CIM_SCSIProtocolController	112
Table 92 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity (Driver)	112
Table 93 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity (FCode/BIOS)	113
Table 94 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity (Firmware)	113
Table 95 - SMI Referenced Properties/Methods for CIM_StorageExtent	113
Table 96 - SMI Referenced Properties/Methods for CIM_SystemComponent	114
Table 97 - SMI Referenced Properties/Methods for CIM_SystemDevice (Associates System to AlarmDevice)	114
Table 98 - SMI Referenced Properties/Methods for CIM_SystemDevice	

(Associates controller system to PortController).....	115
Table 99 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to SCSIProtocolController)	115
Table 100 - iSCSI Terminology	116
Table 101 - OperationalStatus Values	119
Table 102 - Supported Profiles for iSCSI Initiator	119
Table 103 - CIM Elements for iSCSI Initiator	120
Table 104 - SMI Referenced Properties/Methods for CIM_BindsTo (TCPProtocolEndpoint to IPProtocolEndpoint)	121
Table 105 - SMI Referenced Properties/Methods for CIM_BindsTo (iSCSIProtocolEndpoint to TCPProtocolEndpoint)	121
Table 106 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Host Hardware RAID Controller)	122
Table 107 - SMI Referenced Properties/Methods for CIM_ControlledBy (Host Hardware RAID Controller)	122
Table 108 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation (EthernetPort to IPProtocolEndpoint).....	122
Table 109 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation (EthernetPort to iSCSIProtocolEndpoint).....	123
Table 110 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity (Host Hardware RAID Controller)	123
Table 111 - SMI Referenced Properties/Methods for CIM_EndpointOfNetworkPipe (Between iSCSI Session and iSCSIProtocolEndpoint)	123
Table 112 - SMI Referenced Properties/Methods for CIM_EndpointOfNetworkPipe (Between iSCSI connection and TCPProtocolEndpoint).....	124
Table 113 - SMI Referenced Properties/Methods for CIM_InstalledSoftwareIdentity (Host Hardware RAID Controller)	124
Table 114 - SMI Referenced Properties/Methods for CIM_NetworkPipeComposition (Host Hardware RAID Controller)	124
Table 115 - SMI Referenced Properties/Methods for CIM_PhysicalPackage (Host Hardware RAID Controller)	125
Table 116 - SMI Referenced Properties/Methods for CIM_PortController (Host Hardware RAID Controller)	125
Table 117 - SMI Referenced Properties/Methods for CIM_Product (Host Hardware RAID Controller)	125
Table 118 - SMI Referenced Properties/Methods for CIM_ProductPhysicalComponent (Host Hardware RAID Controller)	126
Table 119 - SMI Referenced Properties/Methods for CIM_ProtocolControllerForPort (Host Hardware RAID Controller)	126
Table 120 - SMI Referenced Properties/Methods for CIM_Realizes (Host Hardware RAID Controller)	126
Table 121 - SMI Referenced Properties/Methods for CIM_SAPAvailableForElement (Host Hardware RAID Controller)	127
Table 122 - SMI Referenced Properties/Methods for CIM_SCSIProtocolController (Host Hardware RAID Controller)	127
Table 123 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity (Host Hardware RAID Controller)	127
Table 124 - SMI Referenced Properties/Methods for CIM_SystemDevice (to EthernetPort)	128
Table 125 - SMI Referenced Properties/Methods for CIM_SystemDevice (to PortController)	128
Table 126 - SMI Referenced Properties/Methods for CIM_SystemDevice (to ProtocolController)	128
Table 127 - SMI Referenced Properties/Methods for CIM_iSCSIConnection (Host Hardware RAID Controller)	129
Table 128 - SMI Referenced Properties/Methods for CIM_iSCSISession (Host Hardware RAID Controller)	130
Table 129 - CIM Elements for SCSI Multipath Management	140
Table 130 - SMI Referenced Properties/Methods for CIM_ConcreteComponent.....	141

Table 131 - SMI Referenced Properties/Methods for CIM_ConcreteDependency	141
Table 132 - SMI Referenced Properties/Methods for CIM_ElementCapabilities	142
Table 133 - SMI Referenced Properties/Methods for CIM_ElementSettingData	142
Table 134 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity (Driver)	142
Table 135 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity (MP API Plugin)	143
Table 136 - SMI Referenced Properties/Methods for CIM_HostedService	143
Table 137 - SMI Referenced Properties/Methods for CIM_MemberOfCollection	143
Table 138 - SMI Referenced Properties/Methods for CIM_Product	144
Table 139 - SMI Referenced Properties/Methods for CIM_SCSIMultipathConfigurationCapabilities	144
Table 140 - SMI Referenced Properties/Methods for CIM_SCSIMultipathSettings	144
Table 141 - SMI Referenced Properties/Methods for CIM_SCSIPathConfigurationService	145
Table 142 - SMI Referenced Properties/Methods for CIM_SCSITargetPortGroup	146
Table 143 - SMI Referenced Properties/Methods for CIM_ServiceAvailableToElement	146
Table 144 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity (Driver)	147
Table 145 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity (MP API Plugin)	147
Table 146 - CIM Elements for SB Multipath Management	150
Table 147 - SMI Referenced Properties/Methods for SNIA_SBInitiatorTargetLogicalUnitPath	150

INFORMATION TECHNOLOGY – STORAGE MANAGEMENT – Part 7: Host elements

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 24775-7 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 to ISO/IEC 24775-6 and 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 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.

Introduction

The host-based storage part of the Information Technology – Storage Management contains profiles and other clauses for management of host-based storage devices. Host-based storage devices provide storage capabilities to a host computer system. Examples of these devices include Fiber Channel Host Bus Adapters, Serial Attached SCSI Host Bus Adapters, RAID Controllers, JBODs (Just-A-Bunch-Of-Disks) and Operating System-discovered storage resources. The host-based profiles describe the manageability required for each device and the connectivity to the host computer system. The host-based profiles leverage existing subprofiles within this standard, as well as other profiles from the Distributed Management Task Force, where applicable, to create a comprehensive management model.

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 third 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 7: Host elements

1 Scope

This part of ISO/IEC 24775 defines management profiles for autonomous, component and abstract profiles for management of host-based storage devices. The autonomous profiles describe the management of a stand-alone host-based storage entity. The component profiles (or subprofiles) describe management of aspects of host-based storage entities that may be used by other autonomous profiles. Finally, this standard describes abstract profiles that may be used as a basis for creating additional Host-based autonomous profiles.

This part of ISO/IEC 24775 includes autonomous profiles:

- "The Host Discovered Resources Profile

This profile defines the model for the storage devices presented to an operating system running on a host computer system. In addition, this profile describes the map of storage associated to a host-computer system that a client application can discover.

- "The Fibre Channel HBA Profile

This profile defines the model and functions of a Fibre Channel HBA that exports block storage to a host computer system from a SAN device (Fibre Channel switch, array, tape library, etc.).

- iSCSI Initiator Subprofile

This profile defines the model and functions necessary to manage an iSCSI initiator.

Component profiles used by autonomous profiles to describe aspects of host-based storage elements and services. The component profiles (subprofiles) defined in this standard include:

- Host Hardware RAID Controller Profile

This profile defines the model and functions of a host-based RAID controller that exports block storage to a host computer system from locally attached storage devices (internal hard drives, JBODs, etc.)

- Storage HBA Profile

This profile defines the model and functions of a SAS, SATA, SPI, or Fibre Channel HBA that exports storage to a host computer system from a SAN device (Fibre Channel switch, array, tape library, etc.).

- Disk Partition Subprofile

The Disk Partition profile models partition (or slice) configuration services provided by operating systems on some platforms.

- SB Multipath Management Subprofile

The SB Multipath Management Subprofile models paths (connections between host controllers and device ports) for environments supporting the SB (Single Byte) command protocol.

- SCSI Multipath Management Subprofile

The SCSI Multipath Management profile models paths (connections between host controllers, device ports, and logical units) for environments supporting the SCSI command protocol.

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 24775-2:2014, *Information technology – Storage management – Part 4: Common architecture*

ISO/IEC 24775-3:2014, *Information technology – Storage management – Part 3: Common profiles*

ISO/IEC 24775-4:2014, *Information technology – Storage management – Part 4: Block devices*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 24775-2, *Information Technology – Storage Management, Part 2 Common Architecture* and the following apply.

3.1

Fibre Channel

serial I/O bus capable of supporting multiple protocols, including access to open system storage (FCP protocol), access to mainframe storage (**FICON**^{TM1} protocol), and IP

3.2

Host Bus Adapter

HBA

I/O adapter that connects a host I/O bus to a computer's memory system

3.3

Host Computer System

any computer system to which disks, disk subsystems, or file servers are attached and accessible for data storage and I/O

3.4

JBOD

cabinet or enclosure of disks

Note 1 to entry: JBOD is an acronym for Just a Bunch Of Disks.

3.5

Logical Disk

block storage on which file systems are built

Note 1 to entry: A logical disk would be formatted for a particular file system.

3.6

Operating System

OS

software that manages the resources of a host computer system

3.7

RAID

family of techniques for managing multiple disks to deliver desirable cost, data availability, and performance characteristics to host environments

Note 1 to entry: RAID is an acronym for Redundant Array of Independent Disks.

1. FICONTM is an example of a suitable product available commercially. This information is given for the convenience of users of this standard and does not constitute an endorsement of this product by SNIA or any standards organization (such as ISO or IEC).

3.8

Storage Device Enclosure

cabinet or enclosure of storage media devices

EXAMPLE: JBOD

3.9

Storage Volume

unit of capacity served from a block storage device

4 Typographical conventions

4.1 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 4. 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.

4.2 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 Standard. 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 Standards. 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.

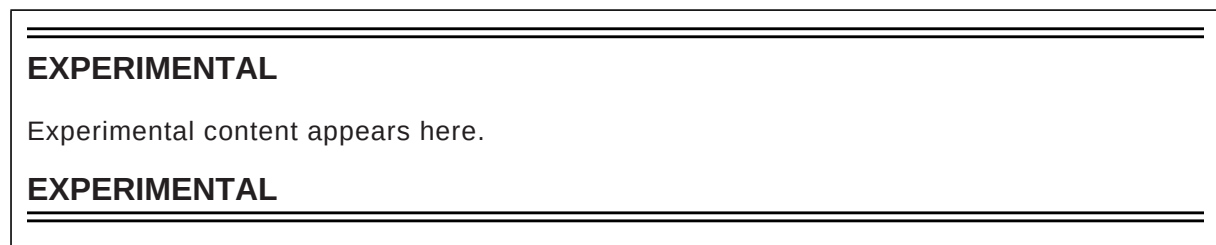


Figure 1 - Experimental maturity level tag

4.3 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.

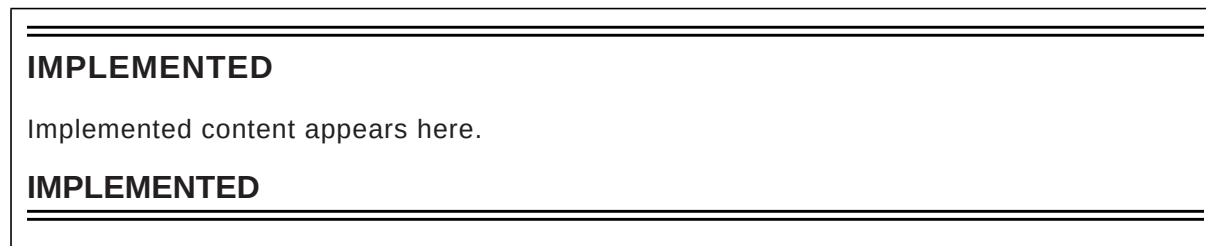


Figure 2 - Implemented maturity level tag

4.4 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.

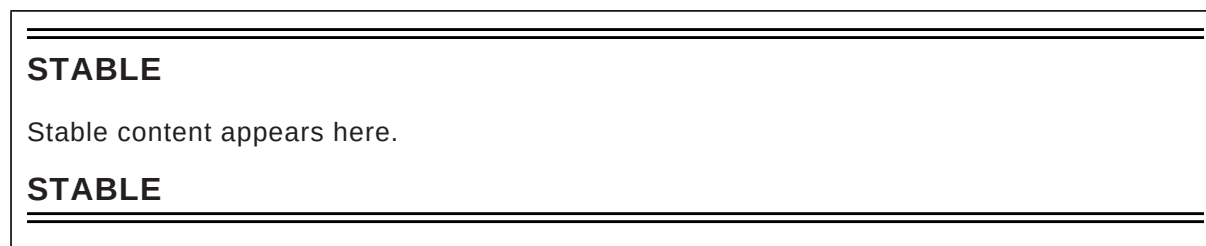


Figure 3 - Stable maturity level tag

4.5 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.

4.6 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 material is

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

EXPERIMENTAL

5 Disk Partition Subprofile

5.1 Description

5.1.1 General

This subprofile models partition (or slice) configuration services provided by operating systems on some platforms. Some operating systems do not use this type of partitioning. On the operating systems that do, the operating system disk drivers treat partitions as virtual disks. The types of valid partitions are determined by the operating system and the partitioning tools.

We need to consider several operating system variants related to operating system partitions.

- On some platforms (e.g., Solaris, Windows), a raw disk volume needs to be partitioned before an application (i.e., a filesystem) uses it. There may be just a single partition on the volume. In these platforms, there is not a name that represents an entire disk volume if that disk volume has multiple partitions.
- On other platforms (e.g., Linux), an application resides on a partition or on the entire disk volume.
- Different operating systems have incompatible partitioning approaches and on-disk data structures (e.g. disk labels or partition tables). This standard refers to these approaches as styles. Each style may be supported by multiple operating systems, and most operating systems support multiple styles. The styles supported in this subprofile are MBR (used on all operating systems running on X86 hardware), vtoc (Solaris and other operating systems with a BSD heritage), and EFI (an emerging style that supports multi-terabyte disk volumes).
- Some styles support multiple tiers of partitions - a partition at one tier may have sub-partitions. On Windows, extended partitions are also a second tier with MBR partitions at each tier.
- Some operating systems utilize two tiers of partitions with different styles at different tiers. For example, BSD-derived Unix variants running on X86 platforms: the lower tier is the X86 BIOS-supported MBR partitions. BSD-style slices can be installed on one of the MBR partitions.
- Some operating systems (AIX, HP_UX) have no equivalent to partitioning.
- Some partition styles have a fixed number of partitions (dependent on the partition type). The user can't create or delete partitions, just adjust the properties of one of the pre-defined partitions.

A partitioned disk volume has an associated partition table. The partition table contains information about the partitions on the disk volume – the starting address, length, and (in some cases) the type of the partition. In certain cases, a partition table can be associated with a partition, allowing multiple tiers of partitions.

In order for storage applications (e.g., logical volume managers, filesystems, databases) to use a disk volume, the operating system provides a name for the volume. These names appear to be filenames but are part of one (or a few) special namespaces managed by the operating system. Windows drive letters and Unix /dev/ directories are examples of the special namespaces. Any extent that is consumable by storage applications is modeled a LogicalDisk. The LogicalDisk.Name property provides this special filename. The exported extent resulting from a partition is a LogicalDisk. On systems that do not require partitions, each usable disk volume has a LogicalDisk instance that models the operating system name. Extents that are not available for storage applications are modeled as StorageExtent (or StorageExtent subclasses other than LogicalDisk) instances and have a name derived from the underlying hardware and partition number.

Operating systems may have different partition styles. The most common style is the MBR (Master Boot Record) style used on x86 PCs. This style supports four primary partitions on a disk volume with an optional second-tier (extended/logical partitions). Solaris uses a style called VTOC that is derived from and similar to BSD partitions. VTOC supports eight partitions.

On Solaris X86, VTOC is installed in one X86 MBR primary partition for compatibility with other x86 operating systems. EFI is a new set of interfaces for x86 64-bit environments and includes a partitioning style. Of particular note is that EFI partitions can exceed the two-terabyte limit associated with other partition styles. So many vendors are migrating towards EFI as an option for supporting larger volumes. This profile includes separate specialized subclasses for MBR, VTOC, and EFI partitions. Their relationship is summarized in Figure 5.

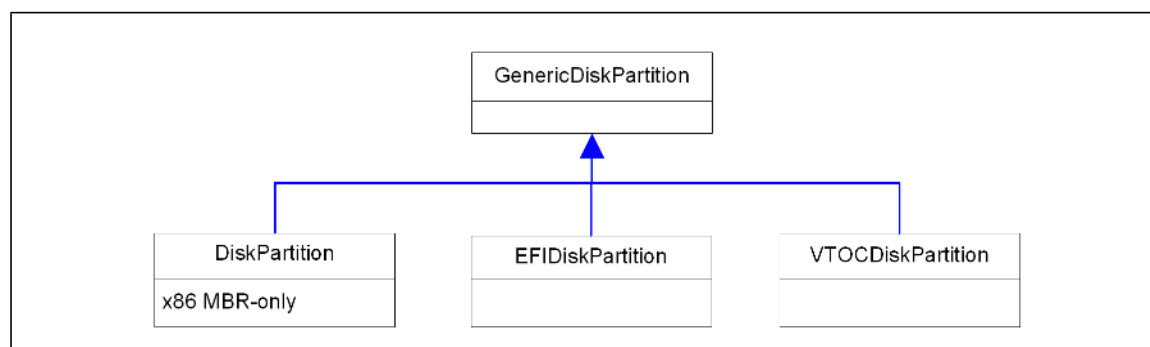


Figure 5 - Disk Partition Class Hierarchy

This profile includes a partition configuration service class that allows a client to create partition tables and modify partitions. It also includes a partition configuration capabilities class that describes the partition configuration capabilities of the system. Separate capabilities instances describe each partition style supported on the system. There should be at most one instance of DiskPartitionConfigurationService, as shown in Figure 6.

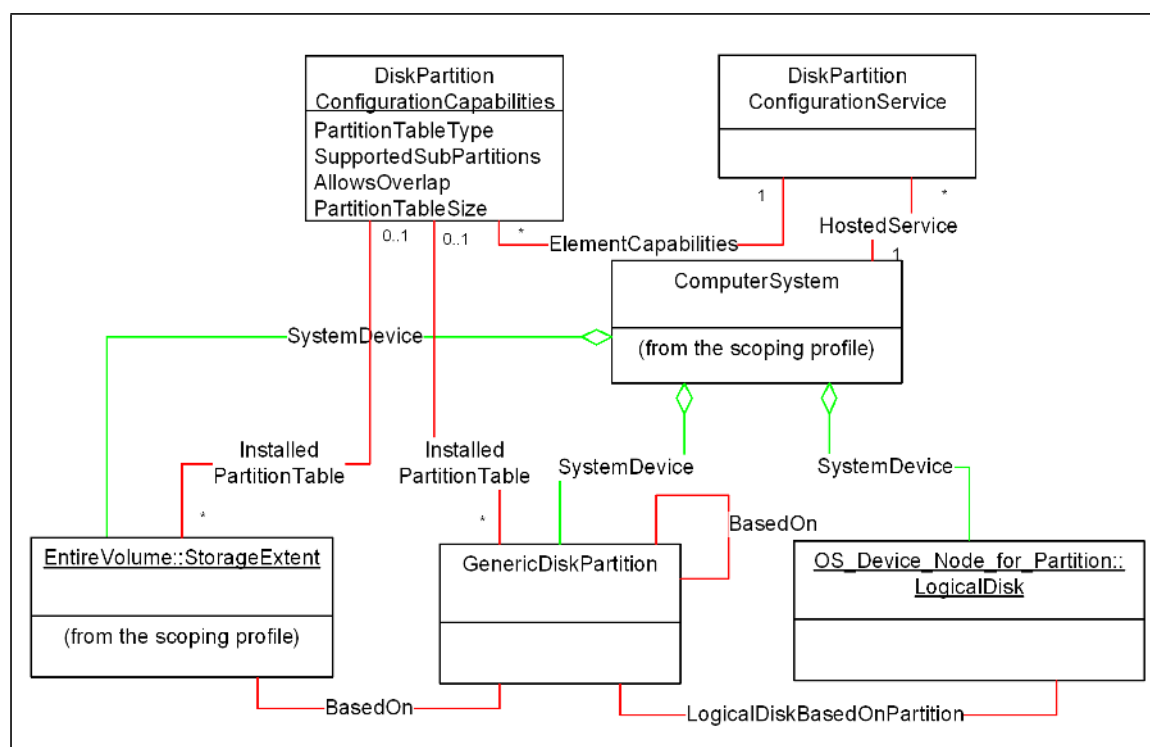


Figure 6 - Disk Partition Class Diagram

5.1.2 Background on X86 MBR Partitions

The terminology used in X86 partition applications is somewhat confusing and masks the actual configurations. The MBR style supports two tiers of partitions: up to four partitions at the entire disk volume tier and up to four partitions within each of these top-tier partitions. An MBR *primary* partition is a top-tier partition that is **not** sub-partitioned. An MBR *extended* partition is

a top-tier partition that **is** sub-partitioned. An MBR *logical* partition is a sub-partition of an extended partition.

Figure 7 represents the actual layout of an MBR drive with three usable partitions – with Windows/DOS driver letter names.

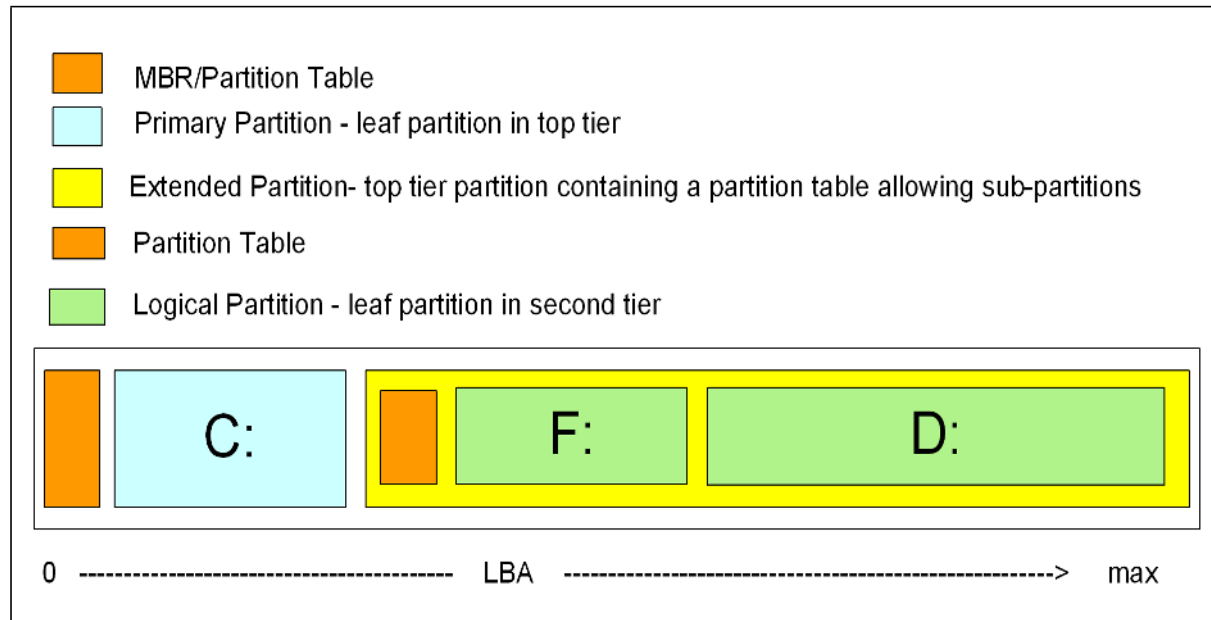


Figure 7 - Disk MBR Partition Example

C: is a primary partition and F: and D: are logical partitions that share an extended partition. Note that the partitions drive letters (C:, F:, and D:) are not in alphabetical order; the assignment of drive letters under Windows/DOS is decoupled from the partitioning logic.

Figure 8 is an instance diagram of the SMI-S classes describing this configuration. Technically, the MBR/Partition tables could be considered to be small partitions. Operating systems generally hide these sectors and treat the effective disk volume as starting just after the MBR. Rather than complicate the SMI-S model, these MBR areas are just ignored and the consumable block size is reduced by the appropriate value (the PartitionTableSize property of DiskPartitionConfigurationCapabilities). In the SMI-S model, the InstalledPartitionTable association to the containing extent indicates the presence of a disk label and/or partition table. In Figure 8, the extent representing the entire disk volume (on the lower left) and the top-tier partition to the right each contain a partition table and are associated to DiskPartitionConfigurationCapabilities via an InstalledPartitionTable association.

In Figure 8 the StorageExtent at the lower left represents the entire disk volume and the two "top-tier" partitions are based on this extent. The LogicalDisk instances at the top represent the consumable partitions C:, F:, and D:.

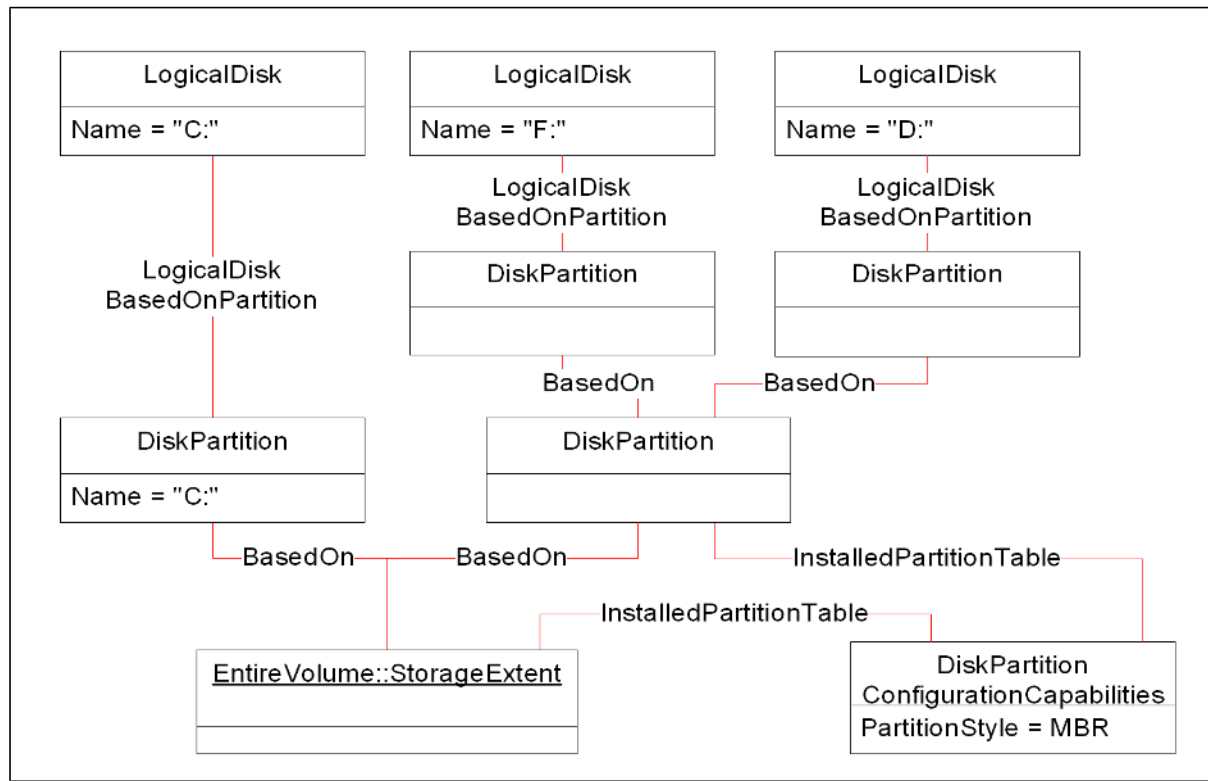


Figure 8 - MBR Partition Instance Diagram

Figure 9 models a similar configuration where the one top-tier partition contains a Solaris X86 installation. In this case, the instrumentation instantiates two instances of DiskPartitionConfigurationCapabilities, one for the top-tier MBR partition table and one for the vtoc partition table.

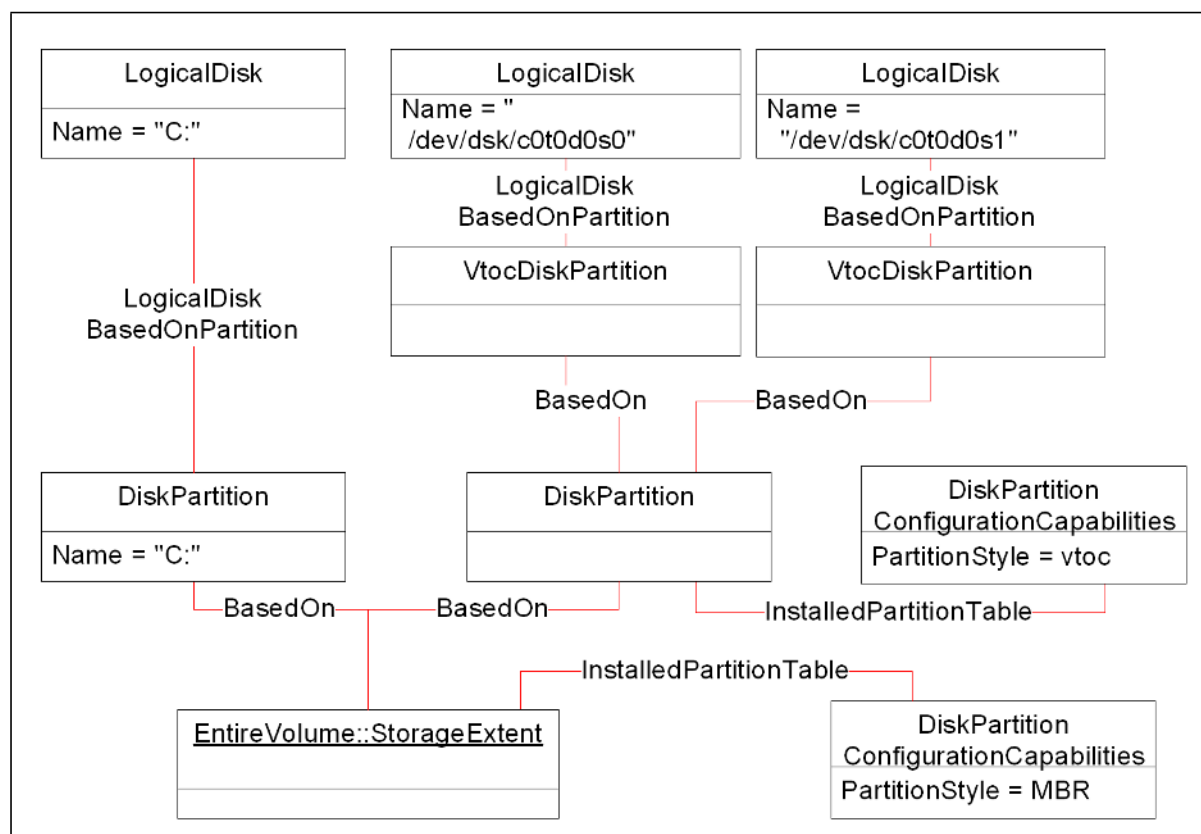


Figure 9 - MBR and VTOC Partition Instance Diagram

Table 1 summarizes likely values for capabilities properties and suggested Name properties on various operating systems.

Table 1 - Capabilities Properties

Property	X86 MBR	vtoc	EFI			
			Win	Linux	Solaris SPARC	Solaris X86
Overlap Allowed	Depends on applications	true	false			
MaxCapacity	2 terabytes (2 ³² blocks)	2 terabytes	2 ⁶⁴ blocks			
MaxNumberOfPartitions	4	8	128	15	127	127

The sizes and starting/ending addresses shall be consistent between the associated LogicalDisk, DiskPartition, and LogicalDisk instances. Figure 10 shows the classes with size information.

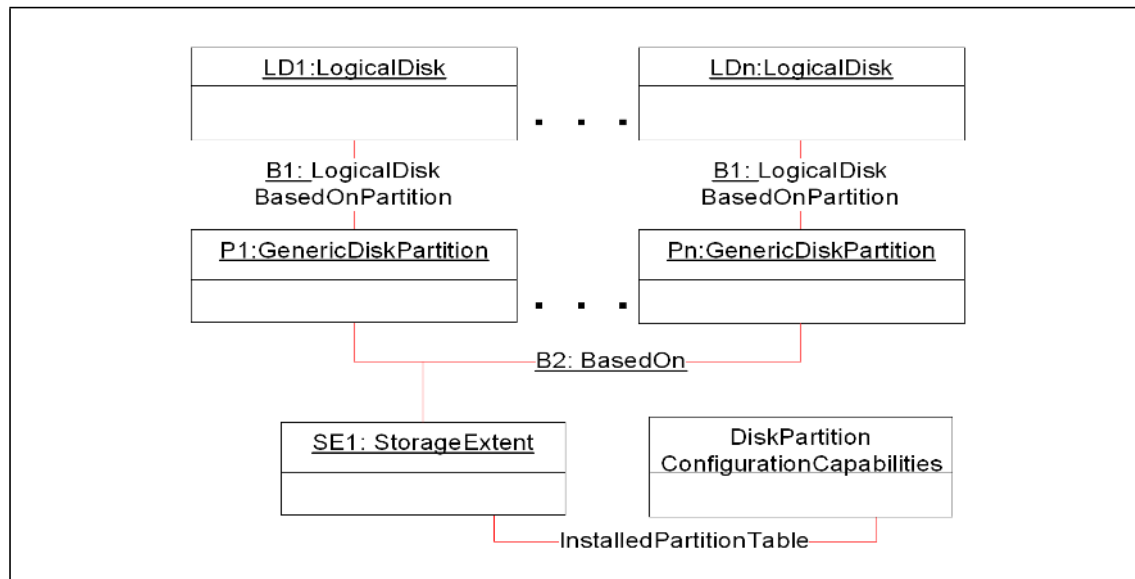


Figure 10 - Partition Instance Diagram for Size/Address Rules

In this diagram, partitions P1,... Pn are all based on the same underlying disk volume (or partition) SE1.

- The NumberOfBlocks shall be the same for a LogicalDisk and its underlying partition (for example, LD1 and P1 in the diagram).
- The StartingAddress in the LogicalDiskBasedOnPartition associations (B1 in the diagram) between a LogicalDisk and its underlying partition will be 0. The EndingAddress in this association shall be one less than NumberOfBlocks from either the LogicalDisk or partition.
- The NumberOfBlocks for each partition (P1, ... Pn) shall be equal to the values of EndingAddress-StartingAddress+1 of the underlying LogicalDiskBasedOnPartition association (B2 in the diagram).
- DiskPartitionConfigurationCapabilities.PartitionTableSize shall hold the total number of blocks consumed by metadata (volume label, boot record, and partition tables) for the associated StorageExtent. For MBR and VTOC styles, this is a fixed value. For EFI, this value could in theory be larger for large extents. Separate instances of DiskPartitionConfigurationCapabilities shall be instantiated as needed to allow different values of PartitionTableSize.
- The size of maintenance tracks or cylinders shall not be included StorageExtent.NumberOfBlocks. This size may be included in DiskPartitionConfigurationCapabilities.PartitionTableSize.
- If DiskPartitionConfigurationCapabilities.OverlapAllowed is false, then the sum of the NumberOfBlocks properties for all partitions plus DiskPartitionConfigurationCapabilities.PartitionTableSize shall not exceed the value of NumberOfBlocks for the underlying StorageExtent. Other than that, there is no guaranteed relationship between StorageExtent.NumberOfBlocks and the sum of the NumberOfBlock values for partitions BasedOn the StorageExtent.

5.2 Health and Fault Management Considerations

No health information is required in LogicalDisk or partition instances. Clients should assume that the health-related properties of the underlying StorageExtent apply to all partitions and LogicalDisks based on that extent.

5.3 Supported Subprofiles and Packages

None.

5.4 Methods of the Profile

5.4.1 SetPartitionStyle

This method installs a partition table on an extent of the specified partition style, creates DiskPartition instances if SettingStyleInstantiatedPartitions is non-zero, and BasedOn associations between the underlying extent and the new partition instances. As a side effect, the usable block size of the underlying extent is reduced by the block size of the metadata reserved by the partition table and associated metadata. This size is in the PartitionTableSize property of the associated DiskPartitionConfigurationCapabilities instance.

```
uint32 SetPartitionStyle (
    [IN, Description (
        "A reference to the extent (volume or partition) where "
        "this style (partition table) will be installed.")]
    CIM_StorageExtent REF Extent,

    [IN, Description (
        "A reference to the "
        "DiskPartitionConfigurationCapabilities instance "
        "describing the desired partition style.")]
    CIM_DiskPartitionConfigurationCapabilities REF PartitionStyle );
```

5.4.2 CreateOrModifyPartition

This method creates a new partition if the Partition parameter is null or modifies the partition specified. If the starting and ending address parameters are null, the resulting partition will occupy the entire underlying extent. If a DeviceFileName parameter is non-null, a LogicalDisk instance is created and associated via LogicalDiskBasedOnPartition to the partition. The underlying extent shall be associated to a capabilities class describing the installed partition style (partition table). This association is established using:

```
uint32 CreateOrModifyPartition (
    [IN, Description (
        "A reference to the underlying extent the partition is "
        "base on.")]
    CIM_StorageExtent REF extent,

    [IN, Description (
        "The starting block number.")]
    uint64 StartingAddress,

    [IN, Description (
        "The ending block number.")]
    uint64 EndingAddress,

    [IN, Description (
        "The platform-specific special file name to be assigned "
        "to the LogicalDisk instance BasedOn the new "
        "DiskPartition instance.")]
    string DeviceFileName,
```

```
[IN, OUT, Description (
    "A reference an existing partition instance to modify or "
    "null to request a new partition.")]
CIM_GenericDiskPartition REF Partition);
```

Delete Instance to delete DiskPartition (and everything south).

5.5 Client Considerations and Recipes

5.5.1 General

A client discovers partition configuration support by looking for instances of DiskPartitionConfigurationService. If no service instances are available, then this operating system does not support disk partitions and the client can assume that any LogicalDisk instance is consumable by applications (such as volume managers or filesystems). For operating systems that do support partitioning, the client can discover whether a particular extent is partitioned by looking for a InstalledPartitionTable instance associated with the extent. The client can discover the existing partition configuration by following BasedOn associations between the extent and GenericDiskPartition instances.

For each discovered service, there shall be one or more instances of DiskPartitionConfigurationCapabilities. There is exactly one capabilities instance per partition table type. If multiple capabilities instances are discovered, the client should look at the SupportedExtentTypes property to determine the services that apply to entire disk volumes and those that apply to partitions.

5.5.2 Create New Partition Using All Available Space at End of Volume

```
//
// Description:
// Create New Partition Using All Available Space at End of Volume
//
// Preconditions:
// $Host holds a ref to the (top-level) ComputerSystem
// $Disk holds a reference to the LogicalDisk (or StorageExtent) instance
// representing the disk or disk volume. $Disk must either be "raw"
// (no volume label), or have some partitioned space at the end.

// Locate instances of CIM_DiskPartitionConfigurationService.
// Note that HDR does not support the multiple system SP,
// so all services must be hosted on $Host.

$Services = AssociatorNames($Host,
    "CIM_HostedService",
    "CIM_DiskPartitionConfigurationService",
    "Antecedent", // Role
    "Dependent") // Result Role
// If no service instances are found, then this platform does
// not support partitioning - so exit.
if ($Service->[].size == 0) {
    <EXIT This system does not support SMI-S disk partitioning>
}

// Look for CIM_DiskPartitionConfigurationCapabilities
// associated to $Disk.
```

```

$Capabilities->[] = AssociatorNames($Host->, // ObjectName
    "CIM_ElementCapabilities", // AssocClass
    "CIM_DiskPartitionConfigurationCapabilities", // ResultClass
    "ManagedElement", // Role
    "Capabilities") // ResultRole

if ($Capabilities != null && $Capabilities->[].size > 1) {
    <ERROR - must not be more than 1
        CIM_DiskPartitionConfigurationCapabilities
        associated with an extent>

#CreateOneBigPartition = false
if ($Capabilities == null || $Capabilities->[].size == 0) {
    // No Capabilities instance found associated to $Disk, this disk has
    // no
    // volume label, create a label with SetPartitionStyle() using
    // the first service instance found.

    // Locate the first Capabilities instance associated with the
    // service. If none, then error.
    $Capabilities->[] =
    // If no capabilities associated to service, then error exit

    %InArguments["Extent"] = $Disk
    %InArguments["Capabilities"] = $Capabilities->[0]
    #MethodReturn = $Services[0]->InvokeMethod(
        $Service->[0],
        "SetPartitionStyle",
        %InArguments)
    if (#MethodReturn != 0) {
        <ERROR - SetPartitionStyle non-zero method return>
    }
    #CreateOneBigPartition = true;
}

// locate partitions based on this disk
$BasedOns[] = References(
    $Disk->[],
    "CIM_BasedOn", // Assoc class
    "Antecedent", // my role
    false,
    false,
    {"StartingAddress", "EndingAddress"})
if ($BasedOns[] == null || $BasedOns->[].size == 0) {
    // If $Disk has no associated partitions, create one using
    // entire disk with CreateOrModifyPartition()
    #CreateOneBigPartition = true;
}

if (#CreateOneBigPartition == true) {
    // null starting and ending address parameters mean

```

```

// "use entire disk".
// null Partition REF parameter means Create
    %InArguments["Extent"] = $Disk
    // all other parms default to "use entire extent"
#MethodReturn = $Services[0]->InvokeMethod(
    $Services->[0],
    "CreateOrModifyPartition",
    %InArguments)
if(#MethodReturn != 0) {
    <ERROR! CreateOrModifyPartition full disk method Failed >
}
}

// Look for available space at end of disk
// Note that the order of partitions in $BasedOns is not necessarily
// the same as the order of the addresses in the partitions.
#CreatePartPossible = true;
// LastBlockInParts is the highest block address in any partition
#LastBlockInParts = $Capabilities.PartitionTableSize
$Capabilities = <get capabilities instance associated with this disk>
for (#i in $BasedOns->[]) {
    // if this partition goes to the end of the underlying extent ...
    if ($BasedOns[#i].EndingAddress == $Disk.NumberOfBlocks-1) {
        // if OverlapAllowed and this partitions takes up entire
        // consumable disk space, then this is a special backup
        // partition - the condition below is the opposite...
        if ((!$Capabilities.OverlapAllowed) ||
            ($BasedOns->[#i].StartingAddress >
             $Capabilities.PartitionTableSize)) {
            #CreatePartPossible = false
        }
    } else {
        // This partition ends after others we've seen (LastBlockInParts)
        // Update LastBlockInParts with the new address
        if ($BasedOns[#i].EndingAddress > #LastBlockInParts) {
            #LastBlockInParts = $BasedOns[#i].EndingAddress
        }
    }
}
if (#CreatePartPossible) {
    if ($BasedOns->[].size() >= $Capabilities.MaxNumberOfPartitions) {
        // then we can't create any more partitions - exit
        exit
    } else {
        // Get the service associated with $Capabilities
        $Service = ..
        $Services = AssociatorNames($Host,
            "CIM_InstalledPartitionTable",
            "CIM_DiskPartitionConfigurationService",
            "Antecedent", // Role

```

```

        "Dependent")// Result Role
        %InArguments["Extent"] = $Disk;
        %InArguments["StartingBlock"] = #LastBlockInParts + 1
        // EndingBlock will default to end of disk
        #MethodReturn = InvokeMethod(
            $Services->[0],
            "CreateOrModifyPartition",
            %InArguments)

        if(#MethodReturn != 0) {
            <ERROR! CreateOrModifyPartition park disk method Failed >
        }
    }
} else {
    <EXIT - no space at end of disk>
}

```

5.6 Registered Name and Version

Disk Partition version 1.3.0 (Component Profile).

5.7 CIM Elements

5.7.1 General

Table 2 describes the CIM elements for Disk Partition.

Table 2 - CIM Elements for Disk Partition

Element Name	Requirement	Description
5.7.2 CIM_BasedOn (Partition to Extent)	Mandatory	
5.7.3 CIM_BasedOn (Partition to Partition)	Mandatory	
5.7.4 CIM_DiskPartitionConfigurationCapabilities	Mandatory	
5.7.5 CIM_DiskPartitionConfigurationService	Mandatory	
5.7.6 CIM_ElementCapabilities	Mandatory	
5.7.7 CIM_GenericDiskPartition	Mandatory	
5.7.8 CIM_HostedService	Mandatory	
5.7.9 CIM_InstalledPartitionTable (Capabilities to Extent)	Mandatory	
5.7.10 CIM_InstalledPartitionTable (Capabilities to Partition)	Mandatory	
5.7.11 CIM_LogicalDisk	Mandatory	
5.7.12 CIM_LogicalDiskBasedOnPartition	Mandatory	
5.7.13 CIM_StorageExtent	Mandatory	
5.7.14 CIM_SystemDevice (System to Extent)	Mandatory	
5.7.15 CIM_SystemDevice (System to LogicalDisk)	Mandatory	
5.7.16 CIM_SystemDevice (System to Partition)	Mandatory	

Table 2 - CIM Elements for Disk Partition

Element Name	Requirement	Description
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_GenericDiskPartition	Mandatory	Partition Creation.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_GenericDiskPartition	Mandatory	Partition Deletion.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_GenericDiskPartition	Mandatory	Partition Modification.

5.7.2 CIM_BasedOn (Partition to Extent)

Created By: Extrinsic: CreateOrModifyPartition

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 3 describes class CIM_BasedOn (Partition to Extent).

Table 3 - SMI Referenced Properties/Methods for CIM_BasedOn (Partition to Extent)

Properties	Flags	Requirement	Description and Notes
StartingAddress		Mandatory	
EndingAddress		Mandatory	
Dependent		Mandatory	
Antecedent		Mandatory	

5.7.3 CIM_BasedOn (Partition to Partition)

Created By: Extrinsic: CreateOrModifyPartition

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 4 describes class CIM_BasedOn (Partition to Partition).

Table 4 - SMI Referenced Properties/Methods for CIM_BasedOn (Partition to Partition)

Properties	Flags	Requirement	Description and Notes
StartingAddress		Mandatory	
EndingAddress		Mandatory	
Antecedent		Mandatory	
Dependent		Mandatory	

5.7.4 CIM_DiskPartitionConfigurationCapabilities

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 5 describes class CIM_DiskPartitionConfigurationCapabilities.

Table 5 - SMI Referenced Properties/Methods for CIM_DiskPartitionConfigurationCapabilities

Properties	Flags	Requirement	Description and Notes
PartitionStyle		Mandatory	
ValidSubPartitionStyles		Mandatory	
MaxNumberOfPartitions		Mandatory	
MaxCapacity		Mandatory	
OverlapAllowed		Mandatory	
PartitionTableSize		Mandatory	

5.7.5 CIM_DiskPartitionConfigurationService

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 6 describes class CIM_DiskPartitionConfigurationService.

Table 6 - SMI Referenced Properties/Methods for CIM_DiskPartitionConfigurationService

Properties	Flags	Requirement	Description and Notes
SetPartitionStyle()		Mandatory	
CreateOrModifyPartition()		Mandatory	

5.7.6 CIM_ElementCapabilities

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 7 describes class CIM_ElementCapabilities.

Table 7 - SMI Referenced Properties/Methods for CIM_ElementCapabilities

Properties	Flags	Requirement	Description and Notes
ManagedElement		Mandatory	
Capabilities		Mandatory	

5.7.7 CIM_GenericDiskPartition

Created By: Extrinsic: CreateOrModifyPartition

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 8 describes class CIM_GenericDiskPartition.

Table 8 - SMI Referenced Properties/Methods for CIM_GenericDiskPartition

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

5.7.8 CIM_HostedService

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 9 describes class CIM_HostedService.

Table 9 - SMI Referenced Properties/Methods for CIM_HostedService

Properties	Flags	Requirement	Description and Notes
Dependent		Mandatory	
Antecedent		Mandatory	

5.7.9 CIM_InstalledPartitionTable (Capabilities to Extent)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 10 describes class CIM_InstalledPartitionTable (Capabilities to Extent).

Table 10 - SMI Referenced Properties/Methods for CIM_InstalledPartitionTable (Capabilities to Extent)

Properties	Flags	Requirement	Description and Notes
Dependent		Mandatory	
Antecedent		Mandatory	

5.7.10 CIM_InstalledPartitionTable (Capabilities to Partition)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 11 describes class CIM_InstalledPartitionTable (Capabilities to Partition).

Table 11 - SMI Referenced Properties/Methods for CIM_InstalledPartitionTable (Capabilities to Partition)

Properties	Flags	Requirement	Description and Notes
Dependent		Mandatory	
Antecedent		Mandatory	

5.7.11 CIM_LogicalDisk

Created By: Extrinsic: CreateOrModifyPartition

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 12 describes class CIM_LogicalDisk.

Table 12 - SMI Referenced Properties/Methods for CIM_LogicalDisk

Properties	Flags	Requirement	Description and Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
DeviceID		Mandatory	
Name		Mandatory	
NameFormat		Mandatory	OS Device Name.
NameNamespace		Mandatory	OS Device Namespace.
OperationalStatus		Mandatory	
NumberOfBlocks		Optional	

5.7.12 CIM_LogicalDiskBasedOnPartition

Created By: Extrinsic: CreateOrModifyPartition

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 13 describes class CIM_LogicalDiskBasedOnPartition.

Table 13 - SMI Referenced Properties/Methods for CIM_LogicalDiskBasedOnPartition

Properties	Flags	Requirement	Description and Notes
Dependent		Mandatory	
Antecedent		Mandatory	

5.7.13 CIM_StorageExtent

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 14 describes class CIM_StorageExtent.

Table 14 - SMI Referenced Properties/Methods for CIM_StorageExtent

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

5.7.14 CIM_SystemDevice (System to Extent)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 15 describes class CIM_SystemDevice (System to Extent).

Table 15 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to Extent)

Properties	Flags	Requirement	Description and Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

5.7.15 CIM_SystemDevice (System to LogicalDisk)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 16 describes class CIM_SystemDevice (System to LogicalDisk).

Table 16 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to LogicalDisk)

Properties	Flags	Requirement	Description and Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

5.7.16 CIM_SystemDevice (System to Partition)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory.

Table 17 describes class Mandatory. CIM_SystemDevice (System to Partition).

Table 17 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to Partition)

Properties	Flags	Requirement	Description and Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

EXPERIMENTAL

IMPLEMENTED

6 FC HBA Profile

6.1 Synopsis

Profile Name: FC HBA (Autonomous Profile).

Version: 1.5.0

Organization: SNIA

CIM Schema Version: 2.9.0

Table 18 describes the related profiles for FC HBA.

Table 18 - Related Profiles for FC HBA

Profile Name	Organization	Version	Requirement	Description
FC Initiator Ports	SNIA	1.4.0	Mandatory	
Software Update	DMTF	1.0.0	Optional	
Indication	SNIA	1.5.0	Mandatory	

Central Class Instance: CIM_ComputerSystem

Scoping Class Instance: CIM_ComputerSystem

The FC HBA Profile describes behavior of Fibre Channel host adapters supporting the SCSI (FCP) command set.

6.2 Description

The FC HBA Profile describes behavior of Fibre Channel host adapters supporting the SCSI (FCP) command set. A Fibre Channel adapter used in a host system is called a Host Bus Adapter (HBA). An HBA is a physical device that contains one or more Fibre Channel ports. A single system contains one or more HBAs.

6.3 Implementation

6.3.1 General

An HBA is represented in CIM by FCPorts associated to a ComputerSystem through the SystemDevice association, as illustrated in Figure 11. To understand the containment to the HBAs physical implementation, the FCPorts are associated to PhysicalPackage through the Realizes association. The PortController represents the logical behavior of the HBA card. It is associated to the ComputerSystem through the SystemDevice association and associated to the FCPorts through the ControlledBy association. PortController's PhysicalPackage is associated with Product, which holds information about the HBA (including vendor and model names).

If the FCPorts reside on a mainboard (rather than a separate card), the same model is used, PortController and PhysicalPackage represent the mainboard. Attributes of Product refer to the vendor and model names of the FCPorts, not the mainboard or system.

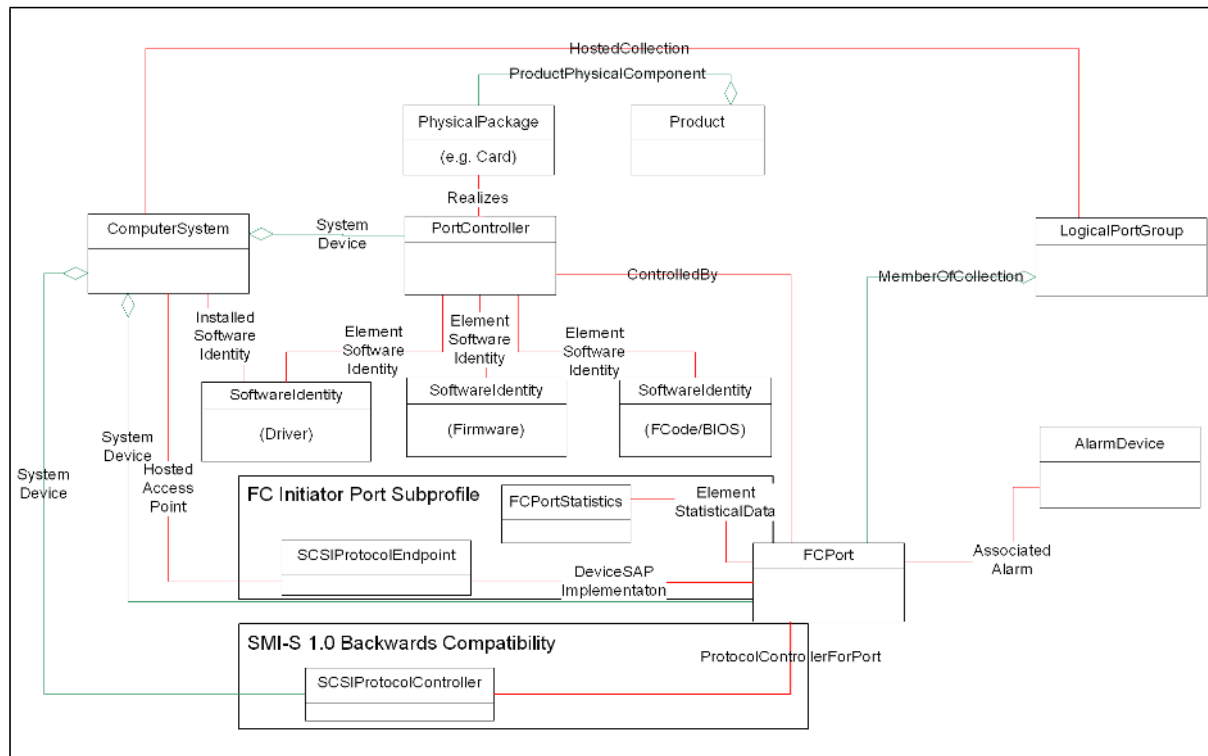


Figure 11 - FC HBA Instance Diagram

Separate instances of SoftwareIdentity represent driver, firmware, and FCODE/BIOS associated with the HBA and includes properties for the vendor, product, and version names (see 6.7 CIM Elements for details). The Classifications property identifies the type (driver, firmware,...). The SoftwareIdentity instance for the driver is mandatory. The others are optional. Note that a separate instance of SoftwareIdentity representing the SMI-S/CIM instrumentation is required by the Profile Registration Profile (ISO/IEC 24775-3, *Information Technology – Storage Management, Part 3 Common Profiles* Clause 42 Profile Registration Profile).

DEPRECATED

6.3.2 Modeling SCSI Protocol Support

The SMI-S 1.0 model for ports and protocols addressed FCP (SCSI over Fibre Channel). As other configurations were considered, the general pattern of initiator port subprofiles (see ISO/IEC 24775-3, *Information Technology – Storage Management, Part 3 Common Profiles*,) emerged. For SMI-S 1.1.0, any initiator port that is configured for SCSI protocol shall use the model in the instance diagram above (ComputerSystem-Hosted Access Point-SCSIProtocolEndpoint-DeviceSAPImplementation-FCPort).

For backwards compatibility, the FC HBA Profile also exposes the SMI-S 1.0 classes (SCSIProtocolController and ProtocolControllerForPort association). In the future, SCSIProtocolController and ProtocolControllerForPort will not be part of the profile. Client applications are encouraged to migrate to the new model.

DEPRECATED

Figure 12 depicts the model for an HBA card with two ports. The LogicalPortGroup represents the collection of ports that shared a Node WWN.

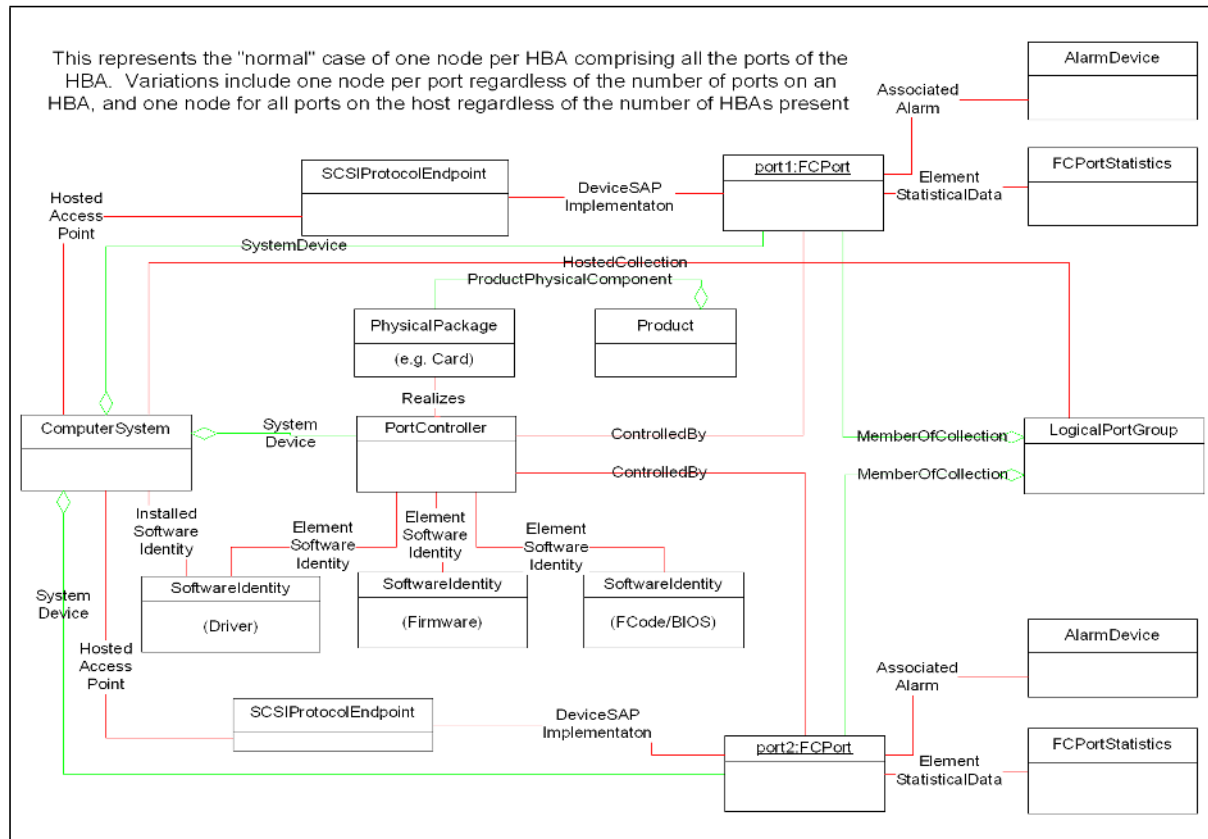


Figure 12 - HBA Card with Two Ports

6.3.3 Persistent Binding

Persistent Binding describes the capability of host adapters to persist user preferences regarding which target logical units are mapped to which OS device names. Persistent Binding for Fibre Channel HBAs is documented in detail in the FC API specification (see <ftp://ftp.t11.org/t11/pub/fc/hba/04-137v0.pdf>).

The term "Persistent Binding" technically refers to the data structure that maps the association from target device correlatable IDs to an OS device name. The collection of these bindings is persisted by the HBA and/or drivers. A persistent binding structure can be defined while the referenced hardware is offline or uninstalled. When the drivers discover attached hardware that matches a persistent binding, the mapping takes place. In many cases, a newly defined persistent binding has no impact until the system is rebooted. The impact will cause target logical units to be attached to initiator SCSIPortals. These associations and target objects are modeled with the Host Discovered Resources Profile.

The persistent binding data structure for bindings that specify OS device names is modeled as OSStorageNameBinding. StorageNameBinding is used when a persistent binding of a device name is determined by the OS. StorageNameBindingService includes methods to create instances of the setting data subclasses, and StorageNameBindingCapabilities provides information about the capabilities of the implementation.

Figure 13 illustrates the Persistent binding model.

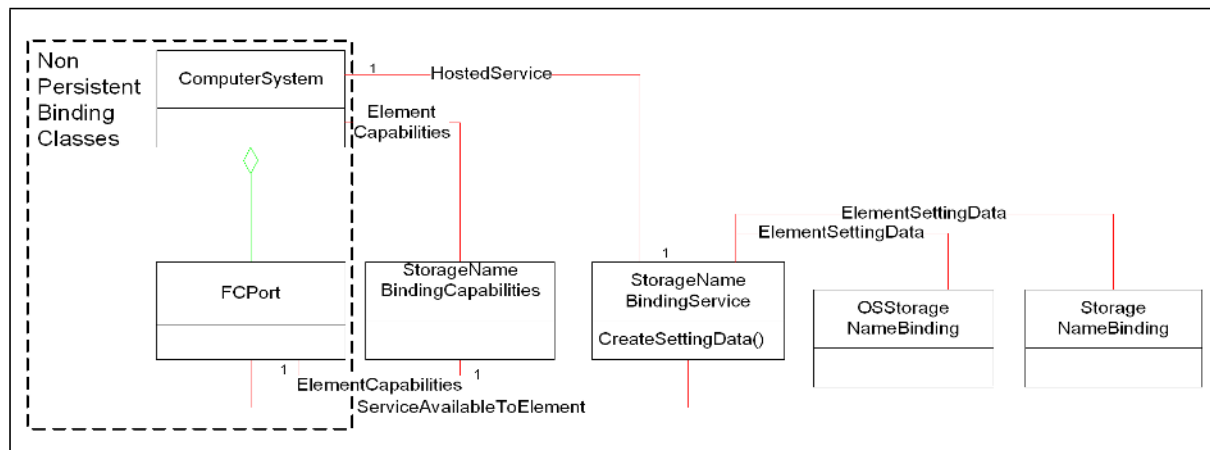


Figure 13 - Persistent Binding Model

Persistent Binding is optional. An implementation that does not support persistent binding (and any of the classes in the diagram above) shall not instantiate an instance of StorageNameBindingService. An implementation that does support persistent binding shall:

- Instantiate a single instance of StorageNameBindingService and associate it to the ComputerSystem.
- Instantiate an instance of StorageNameBindingCapabilities for each FCPort instance, associated via ElementCapabilities.
- At initialization, the implementation shall instantiate instances of OSStorageNameBinding or StorageNameBinding for each previously defined binding.
- Implement the CreateOSStorageNameBinding Method if any StorageNameBindingCapabilities exists with CanSetOSDeviceName set to true.
- Implement the CreateStorageNameBinding Method if StorageNameBindingCapabilities instance exists with CanSetOSDeviceName set to false.
- Support DeleteInstance for StorageNameBinding and OSStorageNameBinding.
- Support ModifyInstance of StorageNameBindingCapabilities.

6.3.4 LED Blink

Implementations may optionally support LED blinking by instantiating an AlarmDevice instance and associating it via AssociatedAlarm to Port instances.

AlarmDevice.VisibleAlarm should be set to true.

AlarmDevice.Urgency should be set to 3 (Informational).

The instrumentation shall provide the SetAlarmState method on AlarmDevice. This method has a single parameter RequestedAlarmState. The only value for this parameter shall be 3 (Alternating).

6.4 Health and Fault Management

Not defined in this standard.

6.5 Methods

6.5.1 Extrinsic Methods of this Profile

The following extrinsic methods are available, but only required if the specific capability (persistent binding or LED blink) is supported.

6.5.2 StorageNameBindingService.CreateStorageNameBinding

This method requests that the driver create a name binding from a target (and optional logical unit) and lets the OS assign the name.

```
uint32 CreateStorageNameBinding (
    [IN, Description ("The value to assign to BindingType."),
    uint16 BindingType,

    [IN, Description ("The value to assign to BindAllLogicalUnits."))
    boolean BindAllLogicalUnits,

    [IN, Description ("The value to assign to Hide."))
    boolean Hide,

    [IN, Description ("The value to assign to TargetName."))
    string TargetName,

    [IN, Description ("The value to assign to LogicalUnitNumber."))
    string LogicalUnitNumber,

    [IN, Description ("The type of the ports in LocalPortNames."),
    // shall be "2" "FC Port WWN"
    uint16 LocalPortNameType,

    [IN, Description ("The values to assign to LocalPortNames."))
    string LocalPortName,

    [IN (false), OUT, Description ("A reference to the created name
    binding instance."))
    StorageNameBinding REF Binding);
```

6.5.3 StorageNameBindingService.CreateOSStorageNameBinding

This method requests that the driver create a name binding from a target (and option logical unit) to a specified OS Device Name or addresses."

```
uint32 CreateOSStorageNameBinding (
    [IN, Description ("The value to assign to BindingType."),
    uint16 BindingType,

    [IN, Description ("The value to assign to BindAllLogicalUnits."))
    boolean BindAllLogicalUnits,

    [IN, Description ("The value to assign to Hide."))
    boolean Hide,

    [IN, Description ("The value to assign to TargetName."))
    string TargetName,
```

```

    [IN, Description ("The value to assign to LogicalUnitNumber.")]
    string LogicalUnitNumber,

    [IN, Description ("The value to assign to OSDeviceName.")]
    string OSDeviceName,

    [IN, Description ("The value to assign to OSAddressesValid.")]
    boolean OSAddressesValid,

    [IN, Description ("The value to assign to OSBusNumber.")]
    uint32 OSBusNumber,

    [IN, Description ("The value to assign to OSTargetNumber.")]
    uint32 OSTargetNumber,

    [IN, Description ("The value to assign to OSLUN.")]
    uint32 OSLUN,

    [IN, Description ("The type of the ports in LocalPortNames."),
    // shall be "2" "FC Port WWN"
    uint16 LocalPortNameType,

    [IN, Description ("The values to assign to LocalPortNames.")]
    string LocalPortName,

    [IN (false), OUT, Description ("A reference to the created name
    binding instance.")]
    CIM_StorageNameBinding REF Binding);

```

6.5.4 CIM_AlarmDevice.SetAlarmState

The instrumentation shall provide the SetAlarmState method on AlarmDevice. This method has a single parameter RequestedAlarmState. The instrumentation shall support a value of 3 (Alternating) for this parameter.

6.5.5 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