Storage Management Technical Specification, Part 5 Fabric

Version 1.3.0, Rev 6

This document has been released and approved by the SNIA. The SNIA believes that the ideas, methodologies and technologies described in this document accurately represent the SNIA goals and are appropriate for widespread distribution. Suggestion for revision should be directed to the Technical Council Managing Director at tcmd@snia.org.

SNIA Technical Position

21 April, 2009

Revision History

Revison 1

Date

4 January, 2007

SCRs Incorporated and other changes

Added placeholders for: (FC-SMIS-SCR00018)

- N Port Virtualization (NPIV)
- Methods for Traceroute and FC Ping
- Additional Alert Indications
- Fabric View Classes Subprofile
- Switch View Classes Subprofile
- FC Security Configuration
- Additional port state/status
- Port numbering on a switch and on a blade

Comments

Formatted for new revision, editorial comments displayed.

Revison 2

Date

14 April 2007

SCRs Incorporated and other changes

Fabric Path Performance Subprofile

- Added missing descriptions to the SMI Referenced Properties Table (FC-SMIS-SCR00021) (3-0-1)

Switch Partitioning profile

- Made this an autonomous profile (FC-SMIS-SCR00026) (4-0-0)

Virtual Fabric profile

- Made this an autonomous profile (FC-SMIS-SCR00026) (4-0-0)

Zone Control subprofile

- Added ActivateZoneServiceWithJob and SessionControlWithJob (FC-SMIS-SCR00023) (2-1-0)

Comments

Only minor editorial work for this revision.

Revison 3

Date

19 June 2007

SCRs Incorporated and other changes

Fabric Profile

- Added more FCPort speeds (FC-SMIS- SCR00033) (3-0-0)

FDMI Profile

- Added more FCPort speeds (FC-SMIS- SCR00033) (3-0-0)

Switch Profile

- Added more FCPort speeds (FC-SMIS- SCR00033) (3-0-0)
- Clarified port numbering (and a requirement for CIM 2.16) (FC-SMIS- SCR00029) (4-0-0)
- Added add NPIV properties (and a requirement for CIM 2.16) (FC-SMIS- SCR00030) (4-0-0)

Extender Profile

- Added more FCPort speeds (FC-SMIS- SCR00033) (3-0-0)

Comments

Editorial notes displayed.

Responses to INCITS editor queries re SMI-S 1.1.0 incorporated as applicable.

Typographical Conventions revised in all books: Revised explanation of Experimental text (per SMIS-120-Errata-SCR00061 - Typographical Conventions), added explanations of Draft and Editorial text.

Revision 4

Date

20 July 2007

SCRs Incorporated and other changes

Fabric View Classes Profile

- Added topology view class (FC-SMIS- SCR00027) (3-0-0)
- Added the switch view class (FC-SMIS- SCR00028) (3-0-0)
- **Promoted to Experimental** the Fabric View Classes Profile (SMIS-130-Draft-SCR00014) (3-0-0)

Switch Profile

- Made mandatory some FCPort statistics (FC-SMIS-SCR00031) (3-0-0)
- Added the property PortAvailability to FCPort (FC-SMIS-SCR00032) (2-0-0)
- **Promoted to Experimental** the Switch Profile content (SMIS-130-Draft-SCR00015) (3-0-0)

Comments

Editorial notes displayed, but the DRAFT material is not.

Revision 5

Date

14 November 2007

SCRs Incorporated and other changes

Clause 6: Zone Control Subprofile (SMIS-120-Errata-SCR00080)

- Updates to HostedService tables in ZoneControl

Comments

Editorial notes and DRAFT material are not displayed.

Revision 6

Date

14 January 2009

SCRs Incorporated and other changes

References to *Storage Management Technical Specification, Part 7 Information Lifecycle Management*, deleted.

Removed test CARDINALITY (SMIS-130-Errata-SCR00001)

Invalid version numbers in supported profiles tables replaced with valid numbers (SMIS-130-Errata-SCR00017)

Updated Fabric Profile (SMIS-130-Errata-SCR00028)

Updated Zone Control & Enhanced Zoning and Enhanced Zone Control Profiles (SMIS-130-Errata-SCR00029)

Comments

Editorial notes and DRAFT material are not displayed.

Suggestion for changes or modifications to this document should be sent to the SNIA Storage Management Initiative Technical Steering Group (SMI-TSG) at http://www.snia.org/feedback/.

INTENDED AUDIENCE

This document is intended for use by individuals and companies engaged in developing, deploying, and promoting interoperable multi-vendor SANs through the SNIA organization.

DISCLAIMER

The information contained in this publication is subject to change without notice. The SNIA makes no warranty of any kind with regard to this specification, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The SNIA shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this specification.

Suggestions for revisions should be directed to http://www.snia.org/feedback/.

Portions of the CIM Schema are used in this document with the permission of the Distributed Management Task Force (DMTF). The CIM classes that are documented have been developed and reviewed by both the Storage Networking Industry Association (SNIA) and DMTF Technical Working Groups. However, the schema is still in development and review in the DMTF Working Groups and Technical Committee, and subject to change.

CHANGES TO THE SPECIFICATION

Each publication of this specification is uniquely identified by a three-level identifier, comprised of a version number, a release number and an update number. The current identifier for this specification is version 1.2.0. Future publications of this specification are subject to specific constraints on the scope of change that is permissible from one publication to the next and the degree of interoperability and backward compatibility that should be assumed between products designed to different publications of this standard. The SNIA has defined three levels of change to a specification:

- Major Revision: A major revision of the specification represents a substantial change to the underlying scope or architecture of the SMI-S API. A major revision results in an increase in the version number of the version identifier (e.g., from version 1.x.x to version 2.x x). There is no assurance of interoperability or backward compatibility between releases with different version numbers.
- Minor Revision: A minor revision of the specification represents a technical change to existing content or an adjustment to the scope of the SMI-S API. A minor revision results in an increase in the release number of the specification's identifier (e.g., from x.1.x to x.2.x). Minor revisions with the same version number preserve interoperability and backward compatibility.
- Update: An update to the specification is limited to minor corrections or clarifications of existing specification content. An update will result in an increase in the third component of the release identifier (e.g., from x.x.1 to x.x.2). Updates with the same version and minor release levels preserve interoperability and backward compatibility.

TYPOGRAPHICAL CONVENTIONS

This specification has been structured to convey both the formal requirements and assumptions of the SMI-S API and its emerging implementation and deployment lifecycle. Over time, the intent is that all content in the specification 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 specification, content is assumed to satisfy these requirements and is referred to as "Finalized". Since much of the evolving specification

content in any given release will not have matured to that level, this specification 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.

Experimental Maturity Level

No material is included in this specification unless its initial architecture has been completed and reviewed. Some content included in this specification 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 specification. 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 specification. 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.

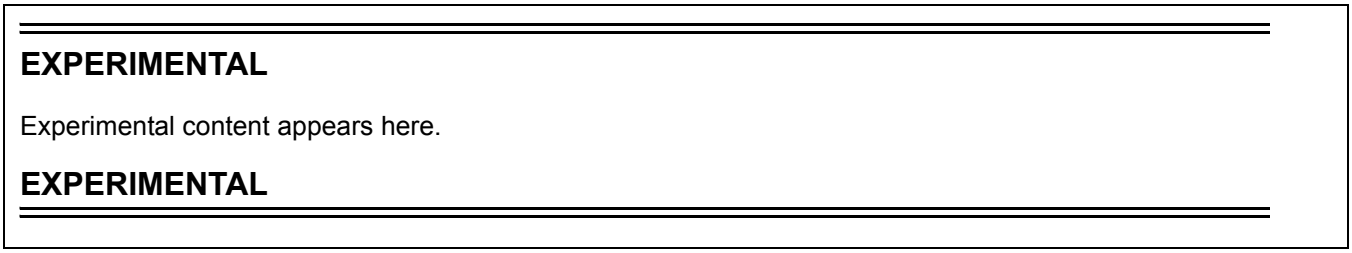

Figure 1 - Experimental Maturity Level Tag

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 must be deprecated in a minor revision of the specification prior to its removal from subsequent releases. Figure 2 is a sample of the typographical convention for Implemented content.

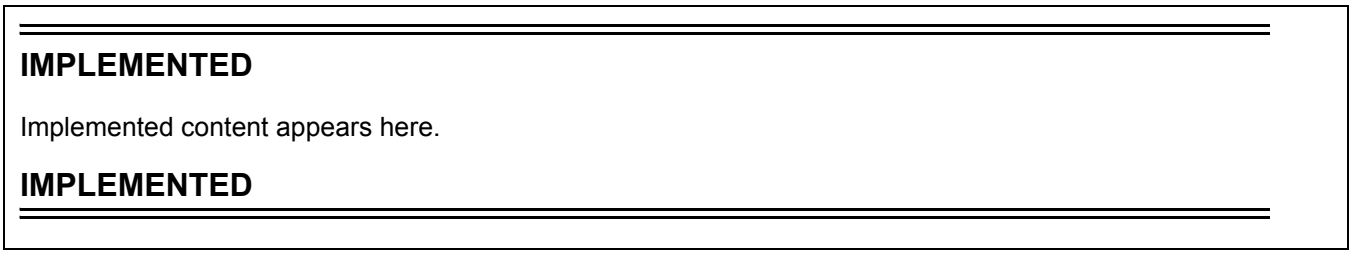

Figure 2 - Implemented Maturity Level Tag

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 specification. Material at this maturity level that has been deprecated may only be removed from the specification as part of a major revision. A profile that has reached this maturity level is guaranteed to preserve backward compatibility from one minor specification 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.

STABLE

Stable content appears here.

STABLE

Figure 3 - Stable Maturity Level Tag

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 must solely depend upon or refine material that has also reached the Finalized level. If specification 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 specification. A profile that has reached this maturity level is guaranteed to preserve backward compatibility from one minor specification revision to the next. Over time, it is hoped that all specification 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 specification 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.

Deprecated Material

Non-Experimental material can be deprecated in a subsequent revision of the specification. 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 specification. 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 specification. Providers shall implement the deprecated elements as long as it appears in the specification 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 specification with the replacement. Figure 4 contains a sample of the typographical convention for deprecated content.

DEPRECATED

Content that has been deprecated appears here.

DEPRECATED

Figure 4 - Deprecated Tag

USAGE

Contents

List of Tables

List of Figures

Foreword

The Fabric Part of the Storage Management Technical Specification defines management profiles for Autonomous (top level) profiles for programs and devices whose central function is providing support for storage networking. The Fabric Part includes fabric management including topology and device management for switches. The Fabric part also provides management of extenders that pass fibre channel frames over other protocols as well as a gateway that maps and translates iSCSI to Fibre Channel. As part of fabric management, this specification also has controls for fibre channel zoning and fibre channel security.

Parts of this Standard

This standard is subdivided in the following parts:

- *Storage Management Technical Specification, Overview, 1.3.0 Rev 6*
- *Storage Management Technical Specification, Part 1 Common Architecture, 1.3.0 Rev 6*
- *Storage Management Technical Specification, Part 2 Common Profiles, 1.3.0 Rev 6*
- *Storage Management Technical Specification, Part 3 Block Devices, 1.3.0 Rev 6*
- *Storage Management Technical Specification, Part 4 File Systems, 1.3.0 Rev 6*
- *Storage Management Technical Specification, Part 5 Fabric, 1.3.0 Rev 6*
- *Storage Management Technical Specification, Part 6 Host Elements, 1.3.0 Rev 6*
- *Storage Management Technical Specification, Part 7 Media Libraries, 1.3.0 Rev 6*

SNIA Web Site

Current SNIA practice is to make updates and other information available through their web site at http://www.snia.org

SNIA Address

Requests for interpretation, suggestions for improvement and addenda, or defect reports are welcome. They should be sent via the SNIA Feedback Portal at http://www.snia.org/feedback/ or by mail to the Storage Networking Industry Association, 500 Sansome Street, Suite #504, San Francisco, CA 94111, U.S.A.

Acknowledgments

The SNIA SMI Technical Steering Group, which developed and reviewed this standard, would like to recognize the significant contributions made by the following members:

Organization Represented	*Name of Representative*
Brocade	John Crandall
Dell	Vance Corn
EMC	Mike Thompson
Hewlett Packard	Alex Lenart
	Steve Peters
Hitachi Data Systems	Steve Quinn
Individual member	Tom West
IBM	Krishna Harathi
	Mike Walker
	Martine Wedlake
Olocity	Scott Baker
Pillar	Gary Steffens
Symantec	Steve Hand
	Paul von Behren

Clause 1: Scope

The Fabric Part of the Storage Management Technical Specification defines management profiles for Autonomous (top level) profiles for programs and devices whose central function is providing support for storage networking.

This version of the Fabric Part of the Storage Management Technical Specification includes four autonomous profiles:

- Fabric

  This profile defines the model and functions of a storage network including topology and zoning control.

- Switch

  This profile defines the model and functions of a Fibre Channel Switch including state, status, and control of the device and it’s connections and product information,

- Extender

  This profile defines the model and functions of a networking device that allows for fibre channel to be extended over other networks ,and specifically over IP (FCIP).

- iSCSI to FC Gateway

  This profile defines the model and functions of a networking device that maps and switches iSCSI frames from a IP network to a fibre channel fabric.

Clause 2: Normative References

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

2.1 Approved references

ISO/IEC 14776-452, SCSI Primary Commands - 2 (SPC-2) [ANSI INCITS.351-2001]

2.2 References under development

Storage Management Technical Specification, Part 1 Common Architecture, 1.3.0 Rev 6

Storage Management Technical Specification, Part 2 Common Profiles, 1.3.0 Rev 6

ISO/IEC 14776-452, SCSI Primary Commands - 3 (SPC-3) [ANSI INCITS 408-2005]

2.3 Other references

DMTF DSP0214:2004 CIM Operations over HTTP

Clause 3: Terms and Definitions

For the purposes of this document, the terms and definitions given in *Storage Management Technical Specification, Part 1 Common Architecture, 1.3.0 Rev 6*.

STABLE

Clause 4: Fabric Profile

4.1 Description

4.1.1 SANS and Fabrics as AdminDomains

A SAN and Fabric are represented in CIM by AdminDomain. A SAN contains one or more Fabrics, which are modeled as AdminDomains. The “containment” of Fabrics to SANs is through the association ContainedDomain. AdminDomain is sub-classed from System. This is significant because a SAN and a Fabric can be considered a group of components that operate together as a single system and should be/are managed as such. The relationship of the Fabrics in a SAN could be as redundant fabrics, interconnected (using the same or different transports/protocols), or not connected in any way. Even in the latter case where the Fabrics are disjoint, from an administrative perspective they may still be managed together applying common practices including naming across the Fabrics.

An AdminDomain in CIM is keyed by the property Name with an associated optional property NameFormat. Typically SANs are identified (“named”) administratively and precise naming conventions are left up to the implementation. The administrator is responsible for assuring that the names are unique within the discovery of known SANs that populate the same CIM Namespace.

For Fibre Channel Fabrics, the identifier (AdminDomain.Name) is the Fabric WWN that is the switch name of the principal switch. The AdminDomain for the Fibre Channel Fabric shall have a NameFormat of WWN.

4.1.2 Fabrics and Topology

A Fabric in CIM minimally contains a ConnectivityCollection and its component systems. They are associated to the Fabric by the association Component. For the purposes of this discussion, it is assumed one models both.

ConnectivityCollection represents the foundation necessary for routing (and is the reason it is defined in the Network model). A ConnectivityCollection groups a set of ProtocolEndpoints that are able to communicate with each other directly. The ProtocolEndpoint is associated to the ConnectivityCollection by MemberOfCollection. A link is represented by the association ActiveCollection, which associates two ProtocolEndpoints, defined as a connection that is currently carrying traffic or is configured to carry traffic.

It is important at this point to clarify the relationship (or use) of the ProtocolEndpoint versus the use of FCPort (discussed later). A NetworkPort (from which FCPort is subclassed) is the device that is used to represent the logical aspects of the link and data layers. The ProtocolEndpoint is used to represent the higher network layers for routing. This is best understood when thinking about Ethernet and IP, but applies to fibre channel also. When two ProtocolEndpoints are capable of communicating, the association ActiveConnection is used to represent the capability to communicate and completes the picture of the topology.

One can ultimately represent multiple ConnectivityCollections (e.g. FC, IP (over FC), and IP (FC encapsulated in IP)) for the same fibre channel fabric.

Note that in modeling SANs, Fabrics, and ConnectivityCollections, a ConnectivityCollection does not require a Fabric, and a Fabric does not require a SAN. But a SAN requires a Fabric, and a Fabric (for the purposes of this profile) requires a ConnectivityCollection.

The minimum set of requirements for this profile is based on FC-GS.

4.1.3 Systems and NetworkPorts

As discussed in the previous section, a Port is associated to a device to represent the link layer. A NetworkPort is associated to the ProtocolEndpoint by DeviceSAPImplementation and “joins” the System and Device model to the Network model. Instantiation of DeviceSAPImplementation, ProtocolEndpoint, and ActiveConnection is not

necessary if the transceiver is not installed or the cable connecting the port to another port is not installed since the device is not capable of communicating.

Systems, or in this case ComputerSystem, represent the fabric elements that contain Ports. These are typically Hosts, Switches and Storage Systems. In Fibre Channel, these are called Platforms and Interconnect Elements. The property Dedicated in ComputerSystem allows these fabric elements to be identified. For a host, Dedicated is set to “Not Dedicated”, for a switch, Dedicated is set to “Switch”, and for a storage system, Dedicated is set to “Storage”. The Ports on a System are associated by SystemDevice.

Discovery from the viewpoint of the fabric includes the end device, but often times the information available is minimal or not available. In the case of Fibre Channel, this occurs if the platform database is not populated. If this is the case, then discovery cannot tell whether a Fibre Channel Node is contained within the same platform or not. When this occurs, ComputerSystem is not instantiated and the LogicalPortGroup representing the Node and the FCPort are associated to the AdminDomain representing the Fabric.

The instrumentation needs to respond to physical fabric changes by adding or removing Logical elements to the AdminDomain. Adding an element to the fabric is straightforward, however it is not always clear when an element has been removed. The device may have been reset, or temporarily shut down, in which case it would be an element in the fabric with an “unknown” status. The lifetime of objects that can no longer be discovered is implementation specific.

If the instrumentation is unable to determine the type of platform discovered (defined in FC-GS), then the agent shall set the ComputerSystem.Dedicated property to “Unknown”.

Additional identification information about ComputerSystem (e.g., DomainID) is placed in OtherIdentifyInfo property.

4.1.4 N Port Virtualization (NPIV)

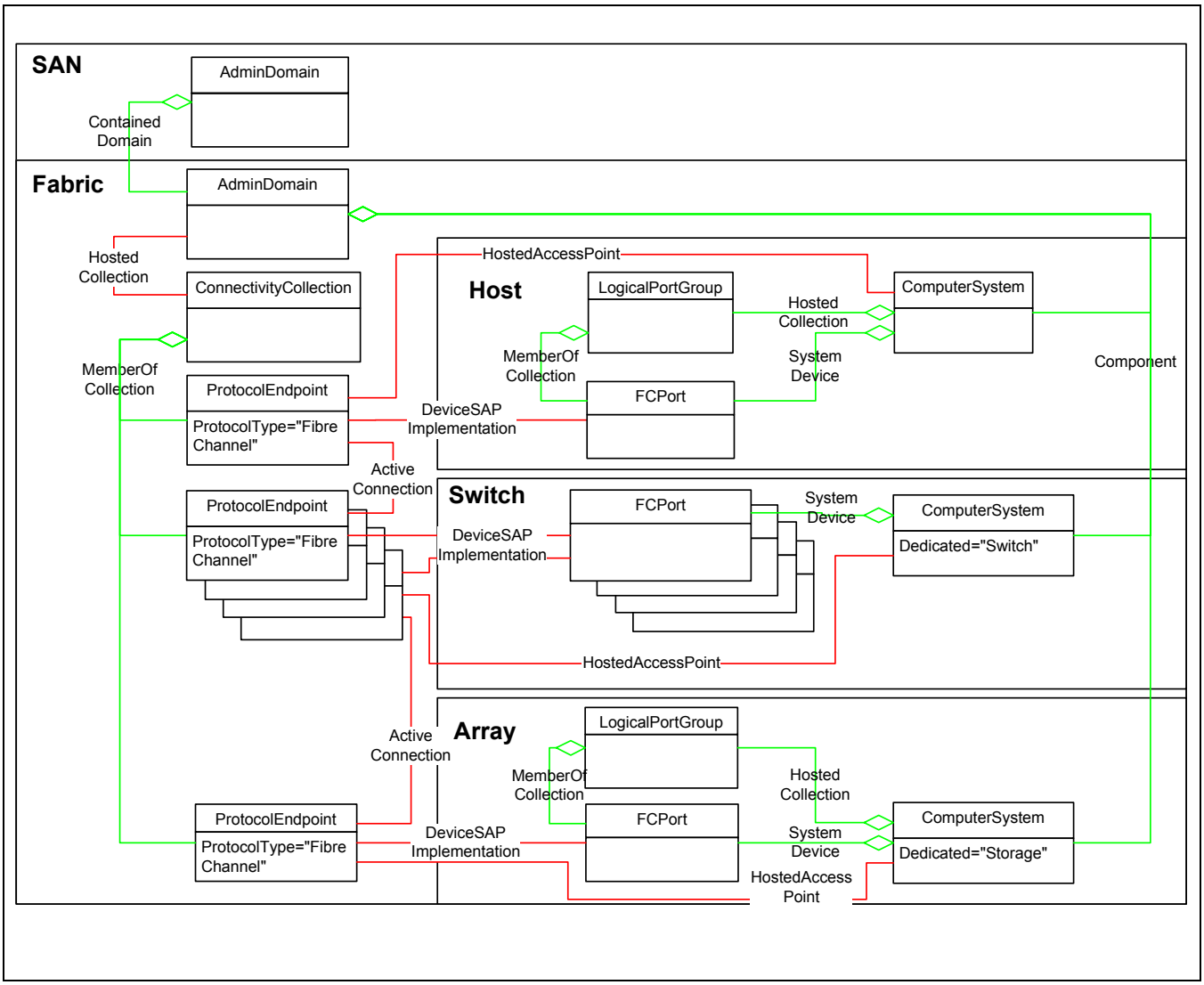

Figure 5 - Fabric Instance

4.1.5 Zoning

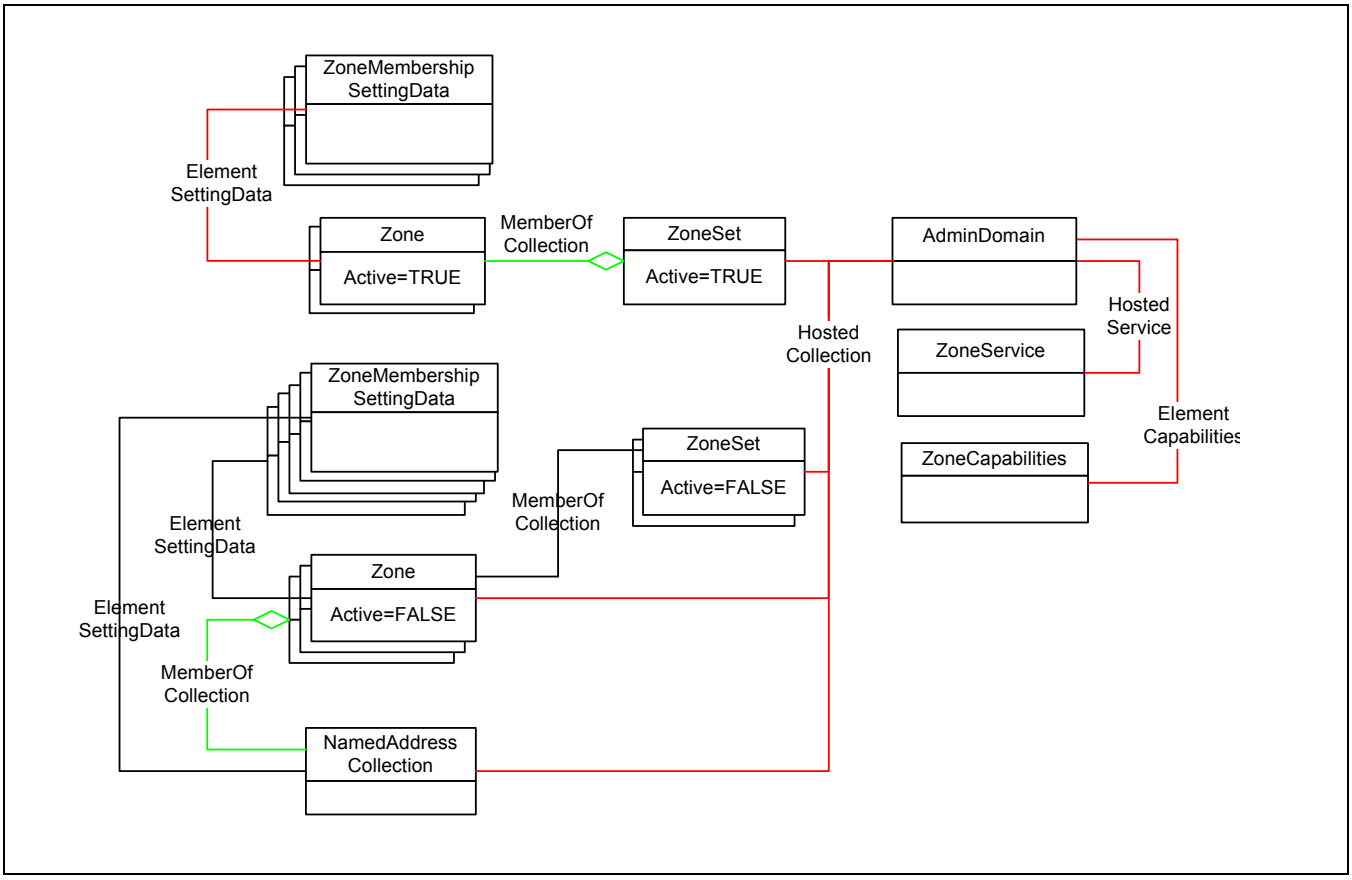

Figure 6 - Zoning Instance (AdminDomain)

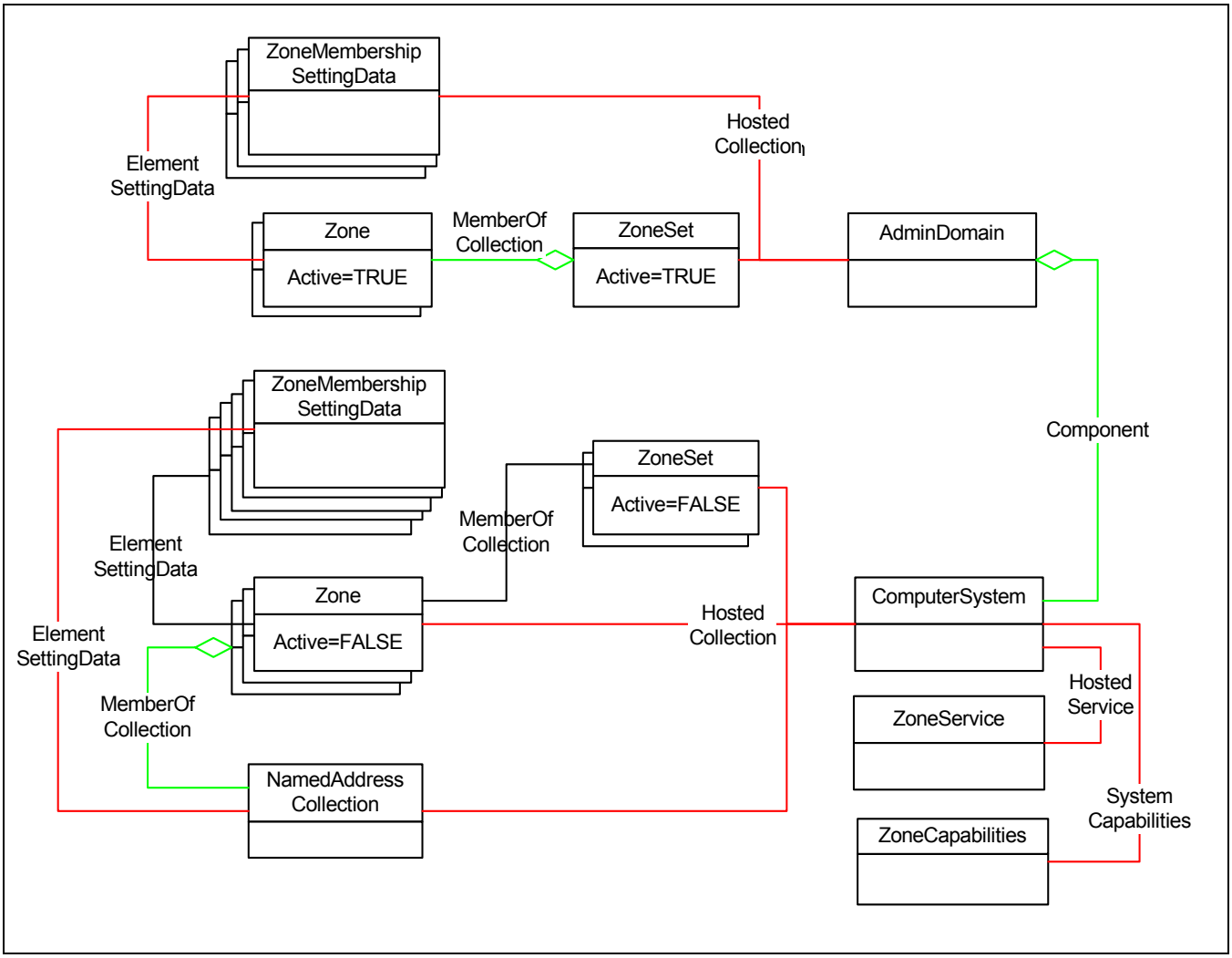

Figure 7 - Zoning Instance (ComputerSystem)

The zoning model is based on FC-GS-4. This model represents the management model for defining Zone Sets, Zones, and Zone Members and “activation” of a Zone Set for a fabric. In the following discussion it may be helpful to also define the following:

- **Active ZoneSet**: the Zone Set currently enforced by the Fabric.
- **Zone Set Database**: The database of the Zone Sets not enforced by the Fabric. Referred to in this document as the Inactive Zone Sets.
- **Zoning Definitions**: a generic term used to indicate both the above concepts.

The zoning model refers to a Zone Set as ZoneSet, a Zone as Zone, ZoneAlias as a NamedAddressCollection, and Zone Member as ZoneMembershipSettingData. ZoneSets shall only contain Zones associated by MemberOfCollection. Zones shall only contain ZoneMembershipSettingData associated by ElementSettingData or NamedAddressCollections associated by MemberOfCollection. For more information with regards to NamedAddressCollection, see Clause 5: Enhanced Zoning and Enhanced Zoning Control Subprofile.

The class ZoneMembershipSettingData has two properties that indicate how the device was identified to be “zoned”. They are ConnectivityMemberType (e.g. PermanentAddress for WWN, NetworkAddress for FCID, etc.) and ConnectivityMemberID which contains the actual device identifier.

The Active Zone Set, defined by an instance of ZoneSet with the Active property set to TRUE, shall only be hosted on the AdminDomain representing the Fabric. The Inactive Zone Sets, defined by an instance of ZoneSet with the Active property set to FALSE, shall be hosted on either the AdminDomain representing the Fabric as shown in Figure 6 or the ComputerSystem representing the switch as shown in Figure 7. It is allowed to have no ZoneSets (active or inactive), only an active ZoneSet, only an inactive ZoneSet(s), or both an inactive ZoneSet(s) and an active ZoneSet.

The ZoneService and ZoneCapabilities are also associated to the same System (AdminDomain or ComputerSystem) as the Inactive Zone Sets using the association HostedService or ElementCapabilities, respectively.

ZoneService provides the configuration methods to control create ZoneSets, Zones, Zone Aliases, and Zone Members, as well as activation of the Zone Set. This service and its methods are described in the Clause 5: Enhanced Zoning and Enhanced Zoning Control Subprofile.

4.2 Health and Fault Management

The following classes report possible Health and Fault information through LifeCycle indications:

- ComputerSystem,
- FCPort

These LifeCycle indications are more fully described in Table 4, “CIM Elements for Fabric,”.

Also in Table 4, “CIM Elements for Fabric,” is a list of AlertIndications which may also be indicators for Health and Fault Management.

4.3 Cascading Considerations

None

4.4 Supported Subprofiles and Package

Table 1 describes the supported profiles for Fabric.

Table 1 - Supported Profiles for Fabric

Registered Profile Names	Mandatory	Version
Zone Control	No	1.3.0
Enhanced Zoning and Enhanced Zoning Control	No	1.3.0
FDMI	No	1.2.0
Fabric Path Performance	No	1.2.0

4.5 Methods of this Profile

4.6 Client Considerations and Recipes

4.6.1 Fabric Identifier

The client needs to consider that the fabric identifier is not durable but is correlatable and may change over time. See *Storage Management Technical Specification, Part 1 Common Architecture, 1.3.0 Rev 6* Clause 7: Correlatable and Durable Names.

4.6.2 FCPort OperationalStatus

OperationalStatus is the property to indicate status and state for the FCPort. The FCPort instance has one of the following Operational Statuses.

Table 2 - Port OperationalStatus

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

4.6.3 ComputerSystem OperationalStatus

OperationalStatus is the property to indicate status and state for the ComputerSystem. The ComputerSystem instance has one of the following Operational Statuses and possibly one of the Subsidiary statuses.

Table 3 - OperationalStatus for ComputerSystem

Operational Status	Possible Subsidiary Operational Status	Description
OK		The system has a good status
OK	Stressed	The system is stressed, for example the temperature is over limit or there is too much IO in progress
OK	Predictive Failure	The system will probably will fail sometime soon
Degraded		The system is operational but not at 100% redundancy. A component has suffered a failure or something is running slow
Error		An error has occurred causing the system to stop. This error may be recoverable with operator intervention.
Error	Non-recoverable error	A severe error has occurred. Operator intervention is unlikely to fix it
Error	Supporting entity in error	A modeled element has failed

Table 3 - OperationalStatus for ComputerSystem

Operational Status	Possible Subsidiary Operational Status	Description
InService		Switch is in Self Test.
No contact		The provider knows about the array but has not talked to it since last reboot
Lost communication		The provider used to be able to communicate with the array, but has now lost contact.
Starting		The system is starting up
Stopping		The system is shutting down.
Stopped		The data path is OK but shut down, the management channel is still working.

4.6.4 Discover The Fabric Topology

```
// This recipe describes how to build a topology graph of a fabric.
//
// 1. Identifies all the Switches and adds their objects paths and the
// object paths of the FC Ports belonging to these Switches to the $nodes
// array
//
// 2. Creates a suitable Association instance (e.g. a SystemDevice
// Association instance between a Switch and a FC Port), setting its
// GroupComponent and PartComponent. Adds the object path of the
// Association to the $links array
//
// 3. Creates a map of all connected FC Ports (i.e., belonging to Switches
// that are ISL'd together and to Host HBAs and Storage System Front End
// Controllers)
//
// In this map, the FC Ports (i.e., the ones that are connected) are
// cross-connected.
//
// e.g., For a pair of FC Ports, one belonging to a Switch and the other
// belonging to a Host (HBA), the map indexed by the Switch Port WWN returns
// the Host (HBA) FC Port object path and the map indexed by the Host (HBA)
// FC Port WWN returns the Switch FC Port object path.
//
// Similar relationship exists between the pairs of FC Ports where one
// belongs to a Switch and the other belonging belongs to a Storage System
// Front End Controller and for FC Ports each of which belongs to a Switch.
//
// 4. Identifies all the Hosts and adds their objects paths to the $nodes
// array. Note that the object paths of the FC Ports (HBA Ports) belonging
// to these Hosts are already added to the $nodes array in step-3.
```

```
//
// 5. Creates a suitable Association instance (e.g. a SystemDevice
// Association instance between a Host and a FC Port), setting its
// GroupComponent and PartComponent. Adds the object path of the Association
// to the $links array.
//
// 6. Identifies all the Storage Systems and adds their objects paths to the
// $nodes array.
// Note that the object paths of the FC Ports (i.e., Front End Controller
// FC Ports) belonging to these Storage Systems are already added to the
// $nodes array in step-3.
//
// 7. Creates a suitable Association instance (e.g. a SystemDevice
// Association instance between a Storage System and a FC Port), setting
// its GroupComponent and PartComponent. Adds the object path of the
// Association to the $links array.

// DESCRIPTION
// Create a map of how elements in a SAN are connected together via
// Fibre-ChannelFC ports.
//
// The map is built in array $attachedFcPorts->[], where the index is a
// WWN of any device port on the SAN, and the value at that index is
// the object path of the connected Switch or HBA or Storage System FC port.
//
// First find all the switches in a SAN. Get all the FC Ports for each
// switch and get the Attached FC Ports for each Switch FC Port. Save these
// device FC ports in the map described above.

// PREEXISTING CONDITIONS AND ASSUMPTIONS
// 1. All agents/namespaces supporting Fabric Profile previously identified
// using SLP. Do this for each CIMOM supporting Fabric Profile

switches[] = enumerateInstances("CIM_ComputerSystem", true, false, true, true,
                            null)
for #i in $switches[]
{
    if (!contains(5, $switches[#i].Dedicated))
        continue

    // only process switches, not other computer systems

    // Add the switch to the $nodes array

    $nodes.addIfNotAlreadyAdded ($switches[#i].getObjectPath();

    // Get all the SystemDevice associations between this switch and its
    // FC Ports
```

```
$sysDevAssoc[] = ReferenceNames($switches[#i],
                                "CIM_FCPort",
                                "GroupComponent");

// Add these associations to the $links array

for #a in $sysDevAssoc->[]
$links.addIfNotAlreadyAdded ($sysDevAssoc->[#a];

$fcPorts->[] = AssociatorNames(
    $switches[#i].getObjectPath(),
    "CIM_SystemDevice",
    "CIM_FCPort",
    "GroupComponent",
    "PartComponent")
for #j in $fcPorts->[]
{

    // Add the FC Port in $nodes array

    $nodes.addIfNotAlreadyAdded (fcPorts->[#j];

    $protocolEndpoints->[] = AssociatorNames(
        fcPorts->[#j],
        "CIM_DeviceSAPImplementation",
        "CIM_ProtocolEndpoint",
        "Antecedent",
    "Dependent");

    // NOTE - It is possible for this collection to be empty (i.e., ports
    // that are not connected). It is possible for this collection to
    // have more than one element (loops attached to a switch port is the
    // most common example).

    if ($protocolEndpoints->[].length == 0)
        continue

    // Add the Protocol End Point to the nodes array.
    // Currently this recipe is designed to only save one
    // ProtocolEndpoint.

    $nodes.addIfNotAlreadyAdded (protocolEndpoints[0]);

    // Add the associations between the fcPort and the Protocol end point
    // to the links array
```

```
$devSAPImplassoc[]  = ReferenceNames($fcPorts->[#j],
                            "CIM_ProtocolEndpoint",
                            null);
for #a in $devSAPImplassoc->[]
    $links.addIfNotAlreadyAdded ($devSAPImplassoc->[#a];

$attachedProtocolEndpoints->[] = AssociatorNames(
    $protocolEndpoints->[0],
    "CIM_ActiveConnection",
    "CIM_ProtocolEndpoint",
    null, null)

// Add the Attached Protocol End Point to the nodes array

$nodes.addIfNotAlreadyAdded (attachedProtocolEndpoints->[0]);

// Add the associations between the Protocol end point and the
// Attached protocol endpoint to the links array

$actConnassoc[]  = ReferenceNames($protocolEndpoint->[#0],
                          "CIM_ActiveConnection",
                           null);
for #a in $actConnassoc->[]
    $links.addIfNotAlreadyAdded ($actConnassoc->[#a];

// NOTE: role & resultRole are null as the direction of the
// association is not dictated by the specification

// $attachedFcPort is either a device FC port or an ISL'd switch FC
// port from another switch. We store this result is stored (i.e.
// which device FC Port is connected // to which switch FC Port) in
// a suitable data structure for subsequent correlation to ports
// discovered on devices.

for #k in $attachedProtocolEndpoints->[]
{
    $attachedFcPorts->[] = Associators(
        $attachedProtocolEndpoints->[#k],
        "CIM_DeviceSAPImplementation",
        "CIM_FCPort",
        "Dependent",
        "Antecedent",
        false,
        false,
        ["PermanentAddress"])
   $attachedFcPort = $attachedFcPorts[0] // Exactly one member guaranteed
                  by model
```

```
            // Add the attached FC Port to the $nodes array
            if $attachedFcPort != null
                $nodes.addIfNotAlreadyAdded ($attachedFcPort);
        }
    }
}

Determine the active Zone Set in a SAN
// DESCRIPTION
// Traverse from the fabric to all zone sets, looking for
// the active zone set
//
// PREEXISTING CONDITIONS AND ASSUMPTIONS
//
// 1. The fabric of interest (an AdminDomain) has been previously
//   identified and defined in the $Fabric-> variable

$ZoneSets[] = Associators($Fabric->, "CIM_HostedCollection", "CIM_ZoneSet", null,
                          null, false, false, null)

for #i in $ZoneSets[] {
    if ($ZoneSet[#i].Active) {
        // <found active ZoneSet>
        // NOTE - there can be only one active ZoneSet in a fabric, though there
                          may be none
        break
    }
}
```

4.7 Registered Name and Version

Fabric version 1.3.0

4.8 CIM Elements

Table 4 describes the CIM elements for Fabric.

Table 4 - CIM Elements for Fabric

Element Name	Requirement	Description
4.8.1 CIM_ActiveConnection	Mandatory	The association between ProtocolEndpoints representing the links between devices.
4.8.2 CIM_AdminDomain (Fabric)	Mandatory	AdminDomain representing the Fabric.
4.8.3 CIM_AdminDomain (SAN)	Mandatory	AdminDomain representing the SAN.
4.8.4 CIM_Component (Platform to Fabric)	Optional	Aggregates Hosts and Arrays in the AdminDomain that represents the Fabric

Table 4 - CIM Elements for Fabric

Element Name	Requirement	Description
4.8.5 CIM_Component (Switch to Fabric)	Mandatory	Aggregates Switches in the AdminDomain that represents the Fabric
4.8.6 CIM_ComputerSystem (Platform)	Mandatory	The ComputerSystem representing the Platform (e.g. Host and Array).
4.8.7 CIM_ComputerSystem (Switch)	Mandatory	The ComputerSystem representing the Switch.
4.8.8 CIM_ConnectivityCollection	Mandatory	Collects the ProtocolEndpoints of the fabric.
4.8.9 CIM_ContainedDomain	Mandatory	Associates a Fabric to a SAN
4.8.10 CIM_DeviceSAPImplementation (Non-Switch to FCPort)	Mandatory	Associates the FCPort to the ProtocolEndpoint
4.8.11 CIM_DeviceSAPImplementation (Switch to FCPort)	Mandatory	Associates the FCPort to the ProtocolEndpoint
4.8.12 CIM_ElementCapabilities (ZoneCapabilities to fabric.)	Mandatory	Associates ZoneCapabilities to a System
4.8.13 CIM_ElementCapabilities (ZoneCapabilities to switch.)	Mandatory	Associates ZoneCapabilities to a switch system
4.8.14 CIM_ElementSettingData (ZoneMembershipSettingData to Zone)	Mandatory	Associates ZoneMembershipSettingData to the Zone.
4.8.15 CIM_FCPort (Non-Switch FCPort)	Mandatory	Fibre Channel Port for Devices
4.8.16 CIM_FCPort (Switch FCPort)	Mandatory	Fibre Channel Port for Switch
4.8.17 CIM_HostedAccessPoint (AdminDomain to ProtocolEndpoint)	Optional	Associates the ProtocolEndpoint to AdminDomain.
4.8.18 CIM_HostedAccessPoint (ComputerSystem to ProtocolEndpoint)	Mandatory	Associates the ProtocolEndpoint to the hosting ComputerSystem
4.8.19 CIM_HostedCollection (Fabric to ConnectivityCollection)	Mandatory	Associates the ConnectivityCollection to the AdminDomain representing the Fabric.
4.8.20 CIM_HostedCollection (System to LogicalPortGroup)	Mandatory	Associates the LogicalPortGroup to the ComputerSystem representing the platform (host or array) or the Fabric (if platforms are not implemented).
4.8.21 CIM_HostedCollection (Zones or ZoneSets to Fabric)	Mandatory	Associates the ZoneSets and Zones to the AdminDomain representing the Fabric.
4.8.22 CIM_HostedCollection (Zones or ZoneSets to Switch)	Mandatory	Associates the ZoneSets and Zones to the hosting System (the ComputerSystem representing the switch).
4.8.23 CIM_LogicalPortGroup	Mandatory	Fibre Channel Node
4.8.24 CIM_MemberOfCollection (ConnectivityCollection to ProtocolEndpoint)	Mandatory	Associates ConnectivityCollection to ProtocolEndpoint

Table 4 - CIM Elements for Fabric

Element Name	Requirement	Description
4.8.25 CIM_MemberOfCollection (LogicalPortGroup to FCPort)	Mandatory	
4.8.26 CIM_MemberOfCollection (ZoneSet to Zone)	Mandatory	
4.8.27 CIM_ProtocolEndpoint	Mandatory	The endpoint of a link (ActiveConnection).
4.8.28 CIM_SystemDevice (Non-Switch FCPort to Fabric)	Optional	Associates non-Switch FCPorts to the System (Fabric)
4.8.29 CIM_SystemDevice (Non-Switch FCPort to Platform)	Optional	Associates non-Switch FCPorts to the ComputerSystem (Platform)
4.8.30 CIM_SystemDevice (Switch FCPort to Switch)	Mandatory	Associates Switch FCPorts to the ComputerSystem (Switch)
4.8.31 CIM_Zone (Active)	Mandatory	The active Zones being enforced by the Fabric.
4.8.32 CIM_Zone (Inactive)	Mandatory	The inactive Zones being enforced by the Fabric.
4.8.33 CIM_ZoneCapabilities	Mandatory	The Zoning Capabilities of the associated Fabric (or Switch).
4.8.34 CIM_ZoneMembershipSettingData	Mandatory	Defines the zone member
4.8.35 CIM_ZoneSet (Active)	Mandatory	The active ZoneSets being enforced by the Fabric.
4.8.36 CIM_ZoneSet (Inactive)	Mandatory	The inactive ZoneSet
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_InstCreation WHERE SourceInstance ISA CIM_FCPort	Mandatory	Creation of a FC Port instance
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_FCPort	Mandatory	Deletion of a FC Port instance
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.OperationalStatus <> PreviousInstance.OperationalStatus	Mandatory	Deprecated WQL -Modification of OperationalStatus of a FC Port instance
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.CIM_FCPort::OperationalStatus <> PreviousInstance.CIM_FCPort::OperationalStatus	Mandatory	CQL -Modification of OperationalStatus of a FC Port instance

Table 4 - CIM Elements for Fabric

Element Name	Requirement	Description
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND SourceInstance.Operationalstatus <> PreviousInstance.OperationalStatus	Mandatory	Deprecated WQL -Modification of OperationalStatus of a ComputerSystem instance
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND SourceInstance.CIM_ComputerSystem::OperationalStatus <> PreviousInstance.CIM_ComputerSystem::Operationalstatus	Mandatory	CQL -Modification of OperationalStatus of a ComputerSystem instance
SELECT * FROM CIM_AlertIndication WHERE OwningEntity='SNIA' and MessageID='FC1'	Mandatory	CQL -Modification of Zone Database
SELECT * FROM CIM_AlertIndication WHERE OwningEntity=SNIA and MessageID='FC1'	Mandatory	Deprecated WQL -Modification of Zone Database
SELECT * FROM CIM_AlertIndication WHERE OwningEntity='SNIA' and MessageID='FC2'	Mandatory	CQL -ZoneSet Activated
SELECT * FROM CIM_AlertIndication WHERE OwningEntity=SNIA and MessageID='FC2'	Mandatory	Deprecated WQL -ZoneSet Activated

4.8.1 CIM_ActiveConnection

The association between ProtocolEndpoints representing the links between devices (including ISLs). For loops, multiple ActiveConnections are instantiated as one to many relationships.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 5 describes class CIM_ActiveConnection.

Table 5 - SMI Referenced Properties/Methods for CIM_ActiveConnection

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	The reference to the ProtocolEndpoint for one end of the link
Dependent		Mandatory	The reference to the ProtocolEndpoint for the other end of the link

4.8.2 CIM_AdminDomain (Fabric)

AdminDomain representing the fabric. This is a logical entity and can represent virtual fabrics.

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

Table 6 describes class CIM_AdminDomain (Fabric).

Table 6 - SMI Referenced Properties/Methods for CIM_AdminDomain (Fabric)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	Name of Class
Name	C	Mandatory	WWN of Fabric
NameFormat		Mandatory	Shall be 'WWN'.

4.8.3 CIM_AdminDomain (SAN)

AdminDomain representing the SAN.

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

Table 7 describes class CIM_AdminDomain (SAN).

Table 7 - SMI Referenced Properties/Methods for CIM_AdminDomain (SAN)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	Name of Class
Name		Mandatory	An arbitrary name (implementation dependent)
NameFormat		Mandatory	Dependent on the arbitrary name chosen.

4.8.4 CIM_Component (Platform to Fabric)

Aggregates Hosts and Arrays in the AdminDomain that represents the Fabric

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

Table 8 describes class CIM_Component (Platform to Fabric).

Table 8 - SMI Referenced Properties/Methods for CIM_Component (Platform to Fabric)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	Reference to Switch
GroupComponent		Mandatory	Reference to the AdminDomain representing the Fabric

4.8.5 CIM_Component (Switch to Fabric)

Aggregates Switches in the AdminDomain that represents the Fabric

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

Table 9 describes class CIM_Component (Switch to Fabric).

Table 9 - SMI Referenced Properties/Methods for CIM_Component (Switch to Fabric)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	Reference to Switch
GroupComponent		Mandatory	Reference to the AdminDomain representing the Fabric

4.8.6 CIM_ComputerSystem (Platform)

The ComputerSystem representing the Platform (e.g. Host and Array). This class is typically instantiated if the end device has populated the Fibre Channel Platform Database or FDMI.

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

Table 10 describes class CIM_ComputerSystem (Platform).

Table 10 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Platform)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	Name of Class
Name	C	Mandatory	The Platform Name or FDMI Host Name.
ElementName		Mandatory	The Platform Label.

Table 10 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Platform)

Properties	Flags	Requirement	Description & Notes
NameFormat		Mandatory	
Dedicated		Mandatory	For a FC-GS Platform Type of Host, 'Not Dedicated' (0); for storage subsystems, 'Storage' (3); for Gateway, 'Gateway' (20); for Router, 'Router' (4); for Bridge, 'Bridge/Extender' (19); for Platform Type of Other, 'Other' (2).

4.8.7 CIM_ComputerSystem (Switch)

The ComputerSystem representing the Switch.

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

Table 11 describes class CIM_ComputerSystem (Switch).

Table 11 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Switch)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	Name of Class
Name	C	Mandatory	The Switch WWN.
ElementName		Mandatory	The Switch Symbolic Name.
NameFormat		Mandatory	Shall be 'WWN'.
OperationalStatus		Mandatory	One of the defined values (2|3|6|8|9|10|11|12|13) shall be present in the array value.
OtherIdentifyingInfo		Mandatory	DomainID stored in decimal format
Dedicated		Mandatory	Shall be 5 (Switch)
IdentifyingDescriptions		Mandatory	Identifying descriptor for OtherIdentifyingInfo. The value 'DomainID' is in IdentifyingDescriptions and in the corresponding index for OtherIdentifyingInfo the DomainID is placed.

4.8.8 CIM_ConnectivityCollection

Collects the ProtocolEndpoints of the fabric.

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

Table 12 describes class CIM_ConnectivityCollection.

Table 12 - SMI Referenced Properties/Methods for CIM_ConnectivityCollection

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Optional	Not required, can be the Fabric WWN.

4.8.9 CIM_ContainedDomain

Associates one or more Fabrics to a SAN.

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

Table 13 describes class CIM_ContainedDomain.

Table 13 - SMI Referenced Properties/Methods for CIM_ContainedDomain

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	The reference to the AdminDomain representing the Fabric
GroupComponent		Mandatory	The reference to the AdminDomain representing the SAN

4.8.10 CIM_DeviceSAPImplementation (Non-Switch to FCPort)

Associates the FCPort to the ProtocolEndpoint

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

Table 14 describes class CIM_DeviceSAPImplementation (Non-Switch to FCPort).

Table 14 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation (Non-Switch to FCPort)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	Reference to the ProtocolEndpoint
Antecedent		Mandatory	Reference to the FCPort

4.8.11 CIM_DeviceSAPImplementation (Switch to FCPort)

Associates the FCPort to the ProtocolEndpoint

Created By: Static
Modified By: Static

Deleted By: Static
Requirement: Mandatory

Table 15 describes class CIM_DeviceSAPImplementation (Switch to FCPort).

Table 15 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation (Switch to FCPort)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

4.8.12 CIM_ElementCapabilities (ZoneCapabilities to fabric.)

Associates the ZoneCapabilities to a System. The system is the AdminDomain representing the Fabric.

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

Table 16 describes class CIM_ElementCapabilities (ZoneCapabilities to fabric.).

Table 16 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (ZoneCapabilities to fabric.)

Properties	Flags	Requirement	Description & Notes
Capabilities		Mandatory	Reference to ZoneCapabilities
ManagedElement		Mandatory	Reference to AdminDomian

4.8.13 CIM_ElementCapabilities (ZoneCapabilities to switch.)

Associates the ZoneCapabilities to a System. The system normally is the AdminDomain representing the Fabric, but in some cases where the Zone Database is not a fabric entity, it maybe hosted on a ComputerSystem representing the Switch.

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

Table 17 describes class CIM_ElementCapabilities (ZoneCapabilities to switch.).

Table 17 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (ZoneCapabilities to switch.)

Properties	Flags	Requirement	Description & Notes
Capabilities		Mandatory	Reference to ZoneCapabilities
ManagedElement		Mandatory	Reference to ComputerSystem

4.8.14 CIM_ElementSettingData (ZoneMembershipSettingData to Zone)

Associates ZoneMembershipSettingData to the Zone.

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

Table 18 describes class CIM_ElementSettingData (ZoneMembershipSettingData to Zone).

Table 18 - SMI Referenced Properties/Methods for CIM_ElementSettingData (ZoneMembership-SettingData to Zone)

Properties	Flags	Requirement	Description & Notes
SettingData		Mandatory	
ManagedElement		Mandatory	Reference to the Zone

4.8.15 CIM_FCPort (Non-Switch FCPort)

Fibre Channel Port for non-Switches (Storage, Router, Bridge/Extender)

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

Table 19 describes class CIM_FCPort (Non-Switch FCPort).

Table 19 - SMI Referenced Properties/Methods for CIM_FCPort (Non-Switch FCPort)

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	The scoping System's CreationClassName.
SystemName		Mandatory	The scoping System's Name.
CreationClassName		Mandatory	Name of Class
DeviceID		Mandatory	Opaque

Table 19 - SMI Referenced Properties/Methods for CIM_FCPort (Non-Switch FCPort)

Properties	Flags	Requirement	Description & Notes
ElementName		Optional	Port Symbolic Name if available. Otherwise NULL. If the underlying implementation includes characters that are illegal in CIM strings, then truncate before the first of those characters.
PermanentAddress	CD	Optional	Fibre Channel Port WWN. Expressed as 16 un-separated upper case hex digits (see Table 4 for more information about formats).
NetworkAddresses	C	Optional	Fibre Channel ID (FCID). Expressed as 8 un-separated upper case hex digits (see Table 4 for more information about formats).
OperationalStatus		Mandatory	One of the defined values shall be present in the array value.
PortType		Mandatory	The specific port type currently enabled (from FC-GS Port.Type)
LinkTechnology		Mandatory	Shall be 4 ('FC').
SupportedFC4Types		Optional	An array of integers indicating the Fibre Channel FC-4 protocols supported. **EXPERIMENTAL** |When supporting FC-SB, this property shall be either '27' (FC-SB-2 Channel) for Host Ports or '28' (FC-SB-2 Control Unit) for targets.| **EXPERIMENTAL**
SupportedCOS		Optional	An array of integers indicating the Fibre Channel Classes of Service that are supported. **EXPERIMENTAL** |When supporting FC-SB, this property shall be either '2' or '3' for FC-SB-2 Channel or FC-SB-2 Control Unit ports, respectively.| **EXPERIMENTAL**

4.8.16 CIM_FCPort (Switch FCPort)

Fibre Channel Port for Switch

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

Table 20 describes class CIM_FCPort (Switch FCPort).

Table 20 - SMI Referenced Properties/Methods for CIM_FCPort (Switch FCPort)

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	The scoping System's CreationClassName.
SystemName		Mandatory	The scoping System's Name.

Table 20 - SMI Referenced Properties/Methods for CIM_FCPort (Switch FCPort)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	Name of Class
DeviceID		Mandatory	Opaque
ElementName		Mandatory	Port Symbolic Name if available. Otherwise NULL. If the underlying implementation includes characters that are illegal in CIM strings, then truncate before the first of those characters.
PermanentAddress	CD	Mandatory	Fibre Channel Port WWN
Speed		Optional	Speed of zero represents a link not established. 1Gb is 1062500000 bps 2Gb is 2125000000 bps 4Gb is 4250000000 bps 8Gb is 8500000000 bps 10Gb single channel variants are 10518750000 bps 10Gb four channel variants are 12750000000 bps 16Gb is 17000000000 bps 20Gb four channel variants are 25500000000 bps This is the raw bit rate.
OperationalStatus		Mandatory	One of the defined values shall be present in the array value.
PortType		Mandatory	The specific port type currently enabled (from FC-GS Port.Type)
LinkTechnology		Mandatory	'FC'

4.8.17 CIM_HostedAccessPoint (AdminDomain to ProtocolEndpoint)

Associates the ProtocolEndpoint to the AdminDomain for those systems not registered in the Platform Database or discovered through FDMI.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 21 describes class CIM_HostedAccessPoint (AdminDomain to ProtocolEndpoint).

Table 21 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (AdminDomain to ProtocolEndpoint)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	Reference to the ProtocolEndpoint
Antecedent		Mandatory	Reference to the AdminDomain

4.8.18 CIM_HostedAccessPoint (ComputerSystem to ProtocolEndpoint)

Associates the ProtocolEndpoint to the hosting System. The hosting System is a ComputerSystem for the Switch or Platform.

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

Table 22 describes class CIM_HostedAccessPoint (ComputerSystem to ProtocolEndpoint).

Table 22 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (ComputerSystem to ProtocolEndpoint)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	Reference to the ProtocolEndpoint
Antecedent		Mandatory	Reference to the ComputerSystem

4.8.19 CIM_HostedCollection (Fabric to ConnectivityCollection)

Associates the ConnectivityCollection to the AdminDomain representing the Fabric.

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

Table 23 describes class CIM_HostedCollection (Fabric to ConnectivityCollection).

Table 23 - SMI Referenced Properties/Methods for CIM_HostedCollection (Fabric to Connectivity-Collection)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

4.8.20 CIM_HostedCollection (System to LogicalPortGroup)

Associates the LogicalPortGroup to the ComputerSystem representing the platform (host or array) or the Fabric (if platforms are not implemented).

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

Table 24 describes class CIM_HostedCollection (System to LogicalPortGroup).

Table 24 - SMI Referenced Properties/Methods for CIM_HostedCollection (System to LogicalPort-Group)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

4.8.21 CIM_HostedCollection (Zones or ZoneSets to Fabric)

Associates the ZoneSets and Zones to the AdminDomain representing the Fabric.

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

Table 25 describes class CIM_HostedCollection (Zones or ZoneSets to Fabric).

Table 25 - SMI Referenced Properties/Methods for CIM_HostedCollection (Zones or ZoneSets to Fabric)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

4.8.22 CIM_HostedCollection (Zones or ZoneSets to Switch)

Associates the ZoneSets and Zones to the hosting System (the ComputerSystem representing the switch).

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

Table 26 describes class CIM_HostedCollection (Zones or ZoneSets to Switch).

Table 26 - SMI Referenced Properties/Methods for CIM_HostedCollection (Zones or ZoneSets to Switch)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

4.8.23 CIM_LogicalPortGroup

Represents the Fibre Channel Node. Associated to the host system by the HostedCollection Association. The hosting System is either a ComputerSystem representing the Platform or the AdminDomain representing the fabric in the case for those systems not registered in the Platform Database or discovered through FDMI (but available through the Name Server/Management Server).

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 27 describes class CIM_LogicalPortGroup.

Table 27 - SMI Referenced Properties/Methods for CIM_LogicalPortGroup

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
Name	CD	Mandatory	Fibre Channel Node WWN
NameFormat		Mandatory	Shall be 'WWN'.
ElementName	N	Mandatory	Node Symbolic Name if available. Otherwise NULL. If the underlying implementation includes characters that are illegal in CIM strings, then truncate before the first of those characters.

4.8.24 CIM_MemberOfCollection (ConnectivityCollection to ProtocolEndpoint)

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 28 describes class CIM_MemberOfCollection (ConnectivityCollection to ProtocolEndpoint).

Table 28 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (ConnectivityCollection to ProtocolEndpoint)

Properties	Flags	Requirement	Description & Notes
Collection		Mandatory	
Member		Mandatory	

4.8.25 CIM_MemberOfCollection (LogicalPortGroup to FCPort)

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

Table 29 describes class CIM_MemberOfCollection (LogicalPortGroup to FCPort).

Table 29 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (LogicalPortGroup to FCPort)

Properties	Flags	Requirement	Description & Notes
Collection		Mandatory	
Member		Mandatory	

4.8.26 CIM_MemberOfCollection (ZoneSet to Zone)

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

Table 30 describes class CIM_MemberOfCollection (ZoneSet to Zone).

Table 30 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (ZoneSet to Zone)

Properties	Flags	Requirement	Description & Notes
Collection		Mandatory	
Member		Mandatory	

4.8.27 CIM_ProtocolEndpoint

The endpoint of a link (ActiveConnection). ProtocolEndpoint shall be implemented when an ActiveConnection exists. It may be implemented if no ActiveConnections exist.

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

Table 31 describes class CIM_ProtocolEndpoint.

Table 31 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	The scoping System's CreationClassName.
SystemName		Mandatory	The scoping System's Name.
CreationClassName		Mandatory	Name of Class
Name	CD	Mandatory	The Fibre Channel Port WWN.
NameFormat		Mandatory	'WWN'
ProtocolIFType		Mandatory	Shall be 56(Fibre channel).

4.8.28 CIM_SystemDevice (Non-Switch FCPort to Fabric)

Associates non-Switch FCPorts to the System (Fabric)

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

Table 32 describes class CIM_SystemDevice (Non-Switch FCPort to Fabric).

Table 32 - SMI Referenced Properties/Methods for CIM_SystemDevice (Non-Switch FCPort to Fabric)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	The reference to the FcPort
GroupComponent		Mandatory	The reference to the Fabric

4.8.29 CIM_SystemDevice (Non-Switch FCPort to Platform)

Associates non-switch FCPorts to the Platform ComputerSystem.

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

Table 33 describes class CIM_SystemDevice (Non-Switch FCPort to Platform).

Table 33 - SMI Referenced Properties/Methods for CIM_SystemDevice (Non-Switch FCPort to Platform)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	The reference to the FcPort
GroupComponent		Mandatory	The reference to the Platform (Storage or Host) System

4.8.30 CIM_SystemDevice (Switch FCPort to Switch)

Associates the Switch FCPort to the ComputerSystem (Switch).

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

Table 34 describes class CIM_SystemDevice (Switch FCPort to Switch).

Table 34 - SMI Referenced Properties/Methods for CIM_SystemDevice (Switch FCPort to Switch)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	The reference to the switch FCPort
GroupComponent		Mandatory	The reference to the Switch ComputerSystem

4.8.31 CIM_Zone (Active)

The active Zones being enforced by the Fabric.

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

Table 35 describes class CIM_Zone (Active).

Table 35 - SMI Referenced Properties/Methods for CIM_Zone (Active)

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	The Zone Name
ZoneType		Mandatory	The Zone Type
Active		Mandatory	Shall be TRUE. Indicates that this ZoneSet is active.

4.8.32 CIM_Zone (Inactive)

The inactive Zones being enforced by the Fabric.

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

Table 36 describes class CIM_Zone (Inactive).

Table 36 - SMI Referenced Properties/Methods for CIM_Zone (Inactive)

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	The Zone Name
ZoneType		Mandatory	The Zone Type.
Active		Mandatory	Shall be FALSE. Indicates that this ZoneSet is inactive.

4.8.33 CIM_ZoneCapabilities

The Zoning Capabilities of the associated Fabric (or Switch).

ZoneCapabilities exposes the capabilities of the AdminDomain representing the Fabric for active zoning and the capabilities of the ComputerSystem representing the Switch or AdminDomain representing the Fabric for Zone Set Database.

If a ZoneCapability property is not applicable or does not explicitly exists (e.g. the capability is limited only by a memory size), the property is NULL.

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

Table 37 describes class CIM_ZoneCapabilities.

Table 37 - SMI Referenced Properties/Methods for CIM_ZoneCapabilities

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
MaxNumZoneSets		Optional	The maximum number of ZoneSets in the Zone Set Database. NULL should be returned in such cases when the property is not applicable or the number is not limited explicitly.
MaxNumZone		Optional	The maximum number of Zones in the Zone Set Database. NULL should be returned in such cases when the property is not applicable or the number is not limited explicitly.

Table 37 - SMI Referenced Properties/Methods for CIM_ZoneCapabilities

Properties	Flags	Requirement	Description & Notes
MaxNumZoneMembers		Optional	The maximum number of ZoneMembers in the Zone Set Database . All ZoneMembers included in both Zones and ZoneAliases are counted, while the same ZoneMember included in multiple Zones or ZoneAliases is counted only once. NULL should be returned in such cases when the property is not applicable or the number is not limited explicitly.
MaxNumZoneAliases		Optional	The maximum number of ZoneAliases in the Zone Set Database NULL should be returned in such cases when the property is not applicable or the number is not limited explicitly.
ZoneNameMaxLen		Mandatory	The maximum length for the name of a ZoneAlias (NamedAddressCollection.ElementName), Zone (Zone.ElementName) or ZoneSet (ZoneSet.ElementName) the Fabric (or Switch) are capable of supporting.
ZoneNameFormat		Mandatory	The name format of a ZoneAlias *NamedAddressCollection.ElementName), Zone (Zone.ElementName) or ZoneSet (ZoneSet.ElementName) supported by either the Fabric (or the Switch)
MaxNumZonesPerZoneSet		Optional	The maximum number of Zones per ZoneSet. NULL should be returned in such cases when the property is not applicable or the number is not limited explicitly.
SupportedConnectivityMemberTypes		Mandatory	An array containing the supported connectivity member types supported which include Permanent Address (WWN), Switch Port ID (Domain:Port in base10), Network Address (FCID), Logical Port Group (Node WWN).

4.8.34 CIM_ZoneMembershipSettingData

Defines the zone member

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

Table 38 describes class CIM_ZoneMembershipSettingData.

Table 38 - SMI Referenced Properties/Methods for CIM_ZoneMembershipSettingData

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque

Table 38 - SMI Referenced Properties/Methods for CIM_ZoneMembershipSettingData

Properties	Flags	Requirement	Description & Notes
ConnectivityMemberType		Mandatory	Permanent Address (WWN), Switch Port ID (Domain:Port in base10), Network Address (FCID).
ConnectivityMemberID	C	Mandatory	The value of the WWN, Domain/Port, or FCID.

4.8.35 CIM_ZoneSet (Active)

The active ZoneSet being enforced by the Fabric.

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

Table 39 describes class CIM_ZoneSet (Active).

Table 39 - SMI Referenced Properties/Methods for CIM_ZoneSet (Active)

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	The ZoneSet name.
Active		Mandatory	shall be TRUE. Indicates that this ZoneSet is active and members cannot be changed.

4.8.36 CIM_ZoneSet (Inactive)

The inactive ZoneSets.

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

Table 40 describes class CIM_ZoneSet (Inactive).

Table 40 - SMI Referenced Properties/Methods for CIM_ZoneSet (Inactive)

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	The ZoneSet name.
Active		Mandatory	Shall be FALSE. Indicates that this ZoneSet is inactive.

STABLE

STABLE

Clause 5: Enhanced Zoning and Enhanced Zoning Control Subprofile

5.1 Description

This profile describes the additional zoning functions for enhanced zoning. Note that Sessions are normally part of enhanced zoning, but are included in the base fabric profile to address the various types of zoning operations into a single object model. In this subprofile, then only Zone Alias is added.

5.2 Health and Fault Management

None

5.3 Cascading Considerations

None

5.4 Dependencies on Profiles, Subprofiles, and Packages

Table 41 describes the supported profiles for Enhanced Zoning and Enhanced Zoning Control.

Table 41 - Supported Profiles for Enhanced Zoning and Enhanced Zoning Control

Registered Profile Names	Mandatory	Version
Zone Control	Yes	1.3.0

Support for the Clause 6: Zone Control Subprofile is mandatory for the Enhanced Zoning and Enhanced Zoning Control subprofile.

5.5 Methods of this Profile

CreateZoneAlias

The method creates a ZoneAlias and the association HostedCollection. The newly created association, HostedCollection, associates the ZoneAlias to the same AdminDomain the ZoneService is hosted to. For the newly created ZoneAlias, the Active property is always set to false.

```
CreateZoneAlias(
     [IN] string CollectionAlias,
     [OUT] CIM_NamedAddressCollection ref ZoneAlias);
```

AddZoneAlias

Adds to the Zone the specified ZoneAlias.

```
AddZoneAlias(
     [IN] CIM_Zone ref Zone,
     [IN] CIM_NamedAddressCollection ref ZoneAlias);
```

5.6 Client Considerations and Recipes

5.6.1 Create a ZoneAlias

```
// DESCRIPTION
// Create zone alias and add new zone member based on
// the parameters collected by the  CIM Client.
// Before any operations can be imposed on the zoning
// service, a session is requested and obtained from the zone
// service. Create a new ZoneAlias. The session may not be ended if
// the ZoneAlias is empty, so add a zone member to the new ZoneAlias.
// The session is released when the operations are
// completed.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1.The system of interest,either the fabric (AdminDomain)
//   or the switch (ComputerSystem), has been
//   previously identified and defined in the
//   $System-> variable
// 2.The name of the new zone alias is defined in the
//   #ZoneAliasName variable
// 3.   The zone member type is defined in the #ConnectivityMemberType
//   variable
// 4.   The zone member Id of the new zone member is defined in the
//   #ConnectiivityMemberID variable


// 1. Get the ZoneService and start a session
$ZoneServices->[] = AssociatorNames(
     $System->,
     "CIM_HostedService",
     "CIM_ZoneService", null, null)

// Assumption 1 above guarantees there is a zone service for this
// system. the fabric and switch profiles that there is no more than
// one ZoneService for this system
$ZoneService-> = $ZoneServices[0]

if(!&startSession($ZoneService->))
{
     return
}

// 2. Create the ZoneAlias
%InArguments["CollectionAlias"] = #ZoneAliasName
#status = InvokeMethod(
     $ZoneService->,
     "CreateZoneAlias",
```

```
     %InArguments[],
     %OutArguments[])

$ZoneAlias-> = %OutArguments["ZoneAlias"]
if(#status != 0)
     // ERROR!

// 3. Create or locate a ZoneMembershipSettingData
%InArguments["ConnectivityMemberType"] = #ConnectivityMemberType
%InArguments["ConnectivityMemberID"] = #ConnectivityMemberID
%InArguments["SystemSpecificCollection"] = $ZoneAlias->
#status = InvokeMethod($ZoneService->, "CreateZoneMembershipSettingData",
                       %InArguments[], %OutArguments[])

// 4. Add to zone alias if not created as a member of the zone alias
//    Zone member reference is set accordingly in the output arguments.

$ZoneMember-> = %OutArguments["ZoneMembershipSettingData"]

if (#status != 0)
    // ERROR!


// 5. End the session gracefully
&endSession($ZoneService->)
// 6. Verify that the ZoneAlias exists in the database
try{
     GetInstance($ZoneAlias->)
}catch(CIM_ERR_NOT_FOUND){
     // error
}
```

5.6.2 Delete a ZoneAlias

```
// DESCRIPTION
// Delete a zone alias.
// Before any operations can be imposed on the zoning service, a
// session is requested and obtained from the zone service.
// The session is released when the operations are completed.
//
// if the deletion fails, it may be because the Zone Alias is not empty.
// In this case, remove all members from the alias by deleting the
// ElementSettingData associations, and try the deletion again.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1.The system of interest,either the fabric (AdminDomain)
//   or the switch (ComputerSystem), has been
```

```
//   previously identified and defined in the
//   $System-> variable
// 2.The object name of the zone alias to be deleted is
//   defined in the $ZoneAlias-> variable

// 1. Get the zone service and start a session
$ZoneServices->[] = AssociatorNames(
     $System->,
     "CIM_HostedService",
     "CIM_ZoneService",
     null,
     null)

// Assumption 1 above guarantees there is a zone service for this
// system. the fabric and switch profiles that there is no more than
// one ZoneService for this system
$ZoneService-> = $ZoneServices[0]

if(!&startSession($ZoneService->))
{
     return
}

// 2. Attempt to delete the alias
try{
     DeleteInstance($ZoneAlias->)
}catch(CIM_ERR_FAILED){
     // Try to remove any zone members in the alias
     // via the ElementSettingData association
     $ZoneMembers->[] = referenceNames($ZoneAlias->,
            "CIM_ElementSettingData",
            null)
     for #j in $ZoneMembers->[] {
        DeleteInstance(ZoneMembers[#j])
     }
     // Try again
     DeleteInstance($ZoneAlias->)

}
// 3. End Session
&endSession($ZoneService->)
// verify that the deletion occurred
try{
     GetInstance($ZoneAlias->)
}catch(CIM_ERR_NOT_FOUND){
     //expect exception
     return
```

```
}
// error!!
```

5.7 Registered Name and Version

Enhanced Zoning and Enhanced Zoning Control version 1.3.0

5.8 CIM Elements

Table 42 describes the CIM elements for Enhanced Zoning and Enhanced Zoning Control.

Table 42 - CIM Elements for Enhanced Zoning and Enhanced Zoning Control

Element Name	Requirement	Description
5.8.1 CIM_ElementSettingData (ZoneMembershipSettingData to NamedAddressCollection)	Mandatory	Associates ZoneMembershipSettingData to the NamedAddressCollection representing the ZoneAlias.
5.8.2 CIM_HostedCollection (AdminDomain to Collection)	Mandatory	Associates the NameAddressCollection representing the Zone Alias to the AdminDomain
5.8.3 CIM_HostedCollection (ComputerSystem to Collection)	Mandatory	Associates the NameAddressCollection representing the Zone Alias to the System
5.8.4 CIM_MemberOfCollection	Mandatory	Associates NamedAddressCollection with Zone.
5.8.5 CIM_NamedAddressCollection	Mandatory	The Zone Alias.
5.8.6 CIM_ZoneService (Zone Service)	Mandatory	The service that allows for all of the zoning configuration changes.

5.8.1 CIM_ElementSettingData (ZoneMembershipSettingData to NamedAddressCollection)

Associates ZoneMembershipSettingData to the NamedAddressCollection representing the ZoneAlias.

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

Table 43 describes class CIM_ElementSettingData (ZoneMembershipSettingData to NamedAddressCollection).

Table 43 - SMI Referenced Properties/Methods for CIM_ElementSettingData (ZoneMembership-SettingData to NamedAddressCollection)

Properties	Flags	Requirement	Description & Notes
SettingData		Mandatory	
ManagedElement		Mandatory	

5.8.2 CIM_HostedCollection (AdminDomain to Collection)

Associates the NamedAddressCollection representing the Zone Alias to the AdminDomain representing the Fabric.

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

Table 44 describes class CIM_HostedCollection (AdminDomain to Collection).

Table 44 - SMI Referenced Properties/Methods for CIM_HostedCollection (AdminDomain to Collection)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	Reference to the NamedAddressCollection representing the Zone Alias

5.8.3 CIM_HostedCollection (ComputerSystem to Collection)

Associates the NamedAddressCollection representing the Zone Alias to the ComputerSystem representing the switch.

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

Table 45 describes class CIM_HostedCollection (ComputerSystem to Collection).

Table 45 - SMI Referenced Properties/Methods for CIM_HostedCollection (ComputerSystem to Collection)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	Reference to the NamedAddressCollection representing the Zone Alias

5.8.4 CIM_MemberOfCollection

Associates NamedAddressCollection with Zone.

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

Table 46 describes class CIM_MemberOfCollection.

Table 46 - SMI Referenced Properties/Methods for CIM_MemberOfCollection

Properties	Flags	Requirement	Description & Notes
Collection		Mandatory	
Member		Mandatory	

5.8.5 CIM_NamedAddressCollection

The Zone Alias.

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

Table 47 describes class CIM_NamedAddressCollection.

Table 47 - SMI Referenced Properties/Methods for CIM_NamedAddressCollection

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
CollectionAlias		Mandatory	The Zone Alias Name

5.8.6 CIM_ZoneService (Zone Service)

The service that allows for all of the zoning configuration changes. The definition in this profile adds additional methods to the same service defined in Zone Control.

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

Table 48 describes class CIM_ZoneService (Zone Service).

Table 48 - SMI Referenced Properties/Methods for CIM_ZoneService (Zone Service)

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	The scoping System's CreationClassName.
SystemName		Mandatory	The scoping System's Name.
CreationClassName		Mandatory	The Class Name
Name		Mandatory	Opaque
CreateZoneAlias()		Mandatory	
AddZoneAlias()		Mandatory	

STABLE

STABLE

Clause 6: Zone Control Subprofile

6.1 Description

This profile includes extrinsic methods for creating Zone Sets, Zones, and Zone Members and adding Zones to Zone Sets and Zone Members to Zones. Additionally SMI-S defines intrinsic methods for the removing of Zone Members from Zones and Zone Aliases, Zones from Zone Sets, and deleting Zone Members, Zones, and Zone Sets. The profile also includes methods to allow a client to request or release a lock on the fabric for zoning configuration changes.

When an Inactive ZoneSet is “Activated”, new instances representing the Active Zone Set and Active Zones are generated from the Inactive Zone Set definition (where a switch may prune the referenced Zone Set collapsing aliases, removes empty zones, etc.).

When a new Zone Set is “Activated”, the instances representing the previous active Zone Set no longer exists.

This profile also includes the property ConfigurationID on the ZoneService. The ConfigurationID is used to identify a unique point in time identifier of the zoning configuration. It may be a timestamp, a generation identifier, or a checksum. The ConfigurationID shall change whenever the zoning configuration changes. The value of the property is used as an input to ActivateZoneSet() and SessionControl() to allow the client to confirm that the change requested are consistent with the last discovery the client made of the zoning configuration.

In the case where the Inactive Zone Sets are hosted on a switch, the client cannot know which Inactive Zone Set was used to define the current Active Zone Set. Also if two Inactive Zone Sets with the same name are hosted on two different switches, the definitions maybe completely different.

6.2 Durable Names and Correlatable IDs of the Profile

None

6.3 Instrumentation Requirements

The agent shall support the use case defined in 6.8.

6.4 Health and Fault Management

None

6.5 Cascading Considerations

None

6.6 Dependencies on Profiles, Subprofiles, and Packages

Profile Name: Zone Control

Version: 1.3.0

Organization: SNIA

CIM Schema Version: TBD

Related Profiles for Zone Control: Not defined in this standard.

6.7 Methods of this Profile

6.7.1 CreateZoneSet

The method creates a ZoneSet and associates it to the System (AdminDomain representing the Fabric or the ComputerSystem representing the Switch) that the ZoneService is hosted on.

```
CreateZoneSet (
            [IN] string ZoneSetName,
            [OUT] CIM_ZoneSet ref ZoneSet);
```

6.7.2 CreateZone

The method creates a Zone and associates it to System (AdminDomain representing the Fabric or the ComputerSystem representing the Switch) that the ZoneService is hosted on.

```
CreateZone (
            [IN] string ZoneName,
            [IN] uint16 ZoneType,
            [IN] uint16 ZoneSubType,
            [OUT] CIM_Zone ref Zone);
```

6.7.3 CreateZoneMembershipSettingData

The method creates a ZoneMembershipSettingData (a zone member) and adds it to the specified Zone or NamedAddressCollection representing a Zone Alias. The ConnectivityMemberID is dependent upon the ConnectivityMemberType.

For Fibre Channel, the ConnectivityMemberType of “PermanentAddress”, the ConnectivityMemberID is the NxPort WWN; for ConnectivityMemberType of “NetworkAddress”, the ConnectivityMemberID is the NXPort Address ID; for ConnectivityMemberType of “SwitchPortID”, the ConnectivityMemberID is “Domain:PortNumber”.

```
CreateZoneMembershipSettingData (
            [IN] uint16 ConnectivityMemberType,
            [IN] string ConnectivityMemberID,
            [IN] CIM_SystemSpecificCollection ref SystemSpecificCollection,
            [OUT] CIM_ZoneMembershipSettingData ref ZoneMembershipSettingData);
```

6.7.4 AddZone

The method adds to the specified ZoneSet the specified Zone. Adding a Zone to a ZoneSet, extends the zone enforcement definition of the ZoneSet to include the members of that Zone. If adding the Zone is, successful, the Zone should be associated to the ZoneSet by MemberOfCollection.

```
AddZone (
            [IN] CIM_ZoneSet ref ZoneSet,
            [IN] CIM_Zone ref Zone);
```

6.7.5 AddZoneMembershipSettingData

The method adds to the specified Zone or NamedAddessCollection representing the Zone Alias the specified ZoneMembershipSettingData (a zone member).

```
AddZoneMembershipSettingData (
            [IN] CIM_SystemSpecificCollection ref SystemSpecificCollection,
            [IN] CIM_ZoneMembershipSettingData ref ZoneMembershipSettingData);
```

EXPERIMENTAL

6.7.6 ActivateZoneSetWithJob

This method activates the specified ZoneSet with the expectation that a job will be created. It has an optional input, ConfigurationID, which is used to confirm that the ZoneSet being activated matches the definition the application previously discovered or changed. If the ConfigurationID passed in matches the ConfigurationID associated to the current zone definitions, then the fabric attempts to activate the ZoneSet. If the ConfigurationIDs do not match, the method exits without activating the ZoneSet. Once a ZoneSet is activated, a ZoneSet with the property Active set to true, its associated Zones with the property Active set to true, and the Zone's associated ZoneMembershipSettingData are instantiated.

This method may run asynchronously due to the length of the operation. This typically occurs when the session is ended and the changes are committed. When the method is run asynchronously, the value "Method parameters checked - job started" is returned and the output parameter Job is populated with a reference to the job instance.

ActivateZoneSet shall be supported outside of a session. ActivateZoneSet being called within a session is implementation specific.

Calling ActivateZoneSet outside of a session while a session is open is implementation specific.

```
uint32 ActivateZoneSetWithJob (
            [IN] CIM_ZoneSet ref ZoneSet,
            [IN] boolean Activate,
            [IN,OUT] string ConfigurationID,
            [OUT] CIM_ConcreteJob REF Job,
            [IN] datetime TimeoutPeriod)
```

EXPERIMENTAL

6.7.7 ActivateZoneSet

This method activates the specified ZoneSet without a Job. ActivateZoneSetWithJob replaces this method. It is supported for backwards compatibility and should not be used in future implementations.

Once a ZoneSet is activated, a ZoneSet with the property Active set to true, its associated Zones with the property Active set to true, and the Zone's associated ZoneMembershipSettingData are instantiated.

ActivateZoneSet shall be supported outside of a session. ActivateZoneSet being called within a session is implementation specific.

Calling ActivateZoneSet outside of a session while a session is open is implementation specific.

```
uint32 ActivateZoneSet (
            [IN] CIM_ZoneSet ref ZoneSet,
            [IN] boolean Activate )
```

EXPERIMENTAL

6.7.8 SessionControlWithJob

This method allows a client to request or release a lock on the fabric for zoning configuration changes. The method has an optional input, ConfigurationID, which is used to confirm that the ZoneSet being activated matches the

definition the application previously discovered or changed. If the ConfigurationID passed in matches the ConfigurationID associated to the current zone definitions, then the fabric attempts to grant the lock. If the ConfigurationIDs do not match, the method exits without attempting to have the fabric grant the lock.

As described in FC-GS, in the context of Enhanced Zoning Management, management actions to a Zone Server (e.g. write access to the Zoning Database) shall occur only inside a GS session. Clients executing zoning management operations shall use fabric sessions cooperatively if the SMI-S agent supports it. (If the value of SessionState is 4 (“Not Applicable”) then no cooperative session usage is possible).

Before a client executes zoning management operations (intrinsic or extrinsic methods), the client shall request a new session and wait for the request to be granted. To request a new session, first wait until the property “SessionState” of the fabric’s ZoneService is 3 (“Ended”) and the property “RequestedSessionState” is 5 “No Change”. Then call SessionControl with RequestedSessionState = 2 (“Started”). Once zoning management operations are completed, the client shall release the session to enable the provider to propagate changes to the fabric, and to allow other clients to perform management operations. To end a session and commit the changes, call SessionControl with RequestedSessionState = 3 (“Ended”). To abort a sequence of zoning management operations without updating the fabric, call SessionControl with RequestedSessionState = 4 (“Terminated”).

SMI-S agents shall block on calls to SessionControl until the request is fulfilled. For example, an error may occur while committing changes to a fabric, i.e. after a call to SessionControl with RequestedSessionState = 3 (“Ended”). The method cannot return until the session has ended, so that a CIM error can be returned if a problem occurs. While the method is in progress, another client may read the value of the RequestedSessionState property and see the value set by the method currently in progress. Once the request is fulfilled, the RequestedSessionState property is set to value 5 “No Change”, regardless of the value in the setInstance operation.

Sessions can timeout. The session timeout behavior and settings are defined by FC-SW in the section discussing mapping GS sessions for Enhanced Zoning Management.

This method may run asynchronously due to the length of the operation. This shall occur when the session is “Ended” to commit the changes. It shall not be used for any other case. When the method is run asynchronously, the value “Method parameters checked - job started” is returned and the output parameter Job is populated with a reference to the job instance.

A SMIS agent may raise an error if these client cooperation rules are not followed. For the purposes of a SMIS agent, a series of requests from the same authenticated entity are considered to be from a single client. An agent may verify that such a series corresponds to the sequence described above and raise the error CIM_ERR_FAILED at any time if the sequence is violated.

```
uint32 SessionControlWithJob (
            [IN,
             ValueMap {"2", "3", "4"},
             Values {"Started", "Ended", "Terminated"}]
            uint16 RequestedSessionState,
            [IN,OUT] string ConfigurationID,
            [OUT] CIM_ConcreteJob REF Job,
            [IN] datetime TimeoutPeriod)
```

EXPERIMENTAL

6.7.9 SessionControl

The method enables a client to request a lock of the fabric to begin zoning configuration changes. It is supported for legacy implementations. SessionControlWithJob replaces this method. It is supported for backwards compatibility only and should not be used in future implementations.

This method allows a client to request or release a lock on the fabric for zoning configuration changes. As described in FC-GS, in the context of Enhanced Zoning Management, management actions to a Zone Server (e.g.

write access to the Zoning Database) shall occur only inside a GS session. Clients executing zoning management operations shall use fabric sessions cooperatively if the SMI-S agent supports it. (If the value of SessionState is 4 (“Not Applicable”) then no cooperative session usage is possible).

Before a client executes zoning management operations (intrinsic or extrinsic methods), the client shall request a new session and wait for the request to be granted. To request a new session, first wait until the property “SessionState” of the fabric’s ZoneService is 3 (“Ended”) and the property “RequestedSessionState” is 5 “No Change”. Then call SessionControl with RequestedSessionState = 2 (“Started”). Once zoning management operations are completed, the client shall release the session to enable the provider to propagate changes to the fabric, and to allow other clients to perform management operations. To end a session and commit the changes, call SessionControl with RequestedSessionState = 3 (“Ended”). To abort a sequence of zoning management operations without updating the fabric, call SessionControl with RequestedSessionState = 4 (“Terminated”).

SMIS agents shall block on calls to SessionControl until the request is fulfilled. For example, an error may occur while committing changes to a fabric, i.e. after a call to SessionControl with RequestedSessionState = 3 (“Ended”). The method cannot return until the session has ended, so that a CIM error can be returned if a problem occurs. While the method is in progress, another client may read the value of the RequestedSessionState property and see the value set by the method currently in progress. Once the request is fulfilled, the RequestedSessionState property is set to value 5 “No Change”, regardless of the value in the setInstance operation.

Sessions can timeout. The session timeout behavior and settings are defined by INCITS FC-SW in the section discussing mapping GS sessions for Enhanced Zoning Management.

A SMIS agent may raise an error if these client cooperation rules are not followed. For the purposes of a SMIS agent, a series of requests from the same authenticated entity are considered to be from a single client. An agent may verify that such a series corresponds to the sequence described above and raise the error CIM_ERR_FAILED at any time if the sequence is violated.

```
uint32 SessionControl (
            [IN,
             ValueMap {"2", "3", "4"},
             Values {"Started", "Ended", "Terminated"}]
            uint16 RequestedSessionState;};
```

6.7.10 Intrinsics for removing a zone from a zone set

As seen in the instance diagram, a zone is a member of a zone set if there is a “CIM_MemberOfCollection” association from the zone set to the zone. To remove a zone from a zone set, delete the instance of the association “CIM_MemberOfCollection” using the intrinsic operation deleteInstance.

6.7.11 Intrinsics for removing a zone alias from a zone

A zone alias is a member of a zone if there is a “CIM_MemberOfCollection” association from the zone to the zone alias. To remove a zone alias from a zone set, delete the instance of the association “CIM_MemberOfCollection” using the intrinsic operation deleteInstance.

6.7.12 Intrinsics for removing a zone member from a zone or zone alias

Zone members are represented by CIM_ZoneMembershipSettingData instances. No instance of CIM_ZoneMembershipSettingData exists unless it is associated to a zone or zone alias by a CIM_ElementSettingData association. However, an instance of CIM_ZoneMembershipSettingData may be associated to more than one zone or zone alias.

Removing a zone member from a zone or zone alias is equivalent to deleting the instance of the CIM_ElementSettingData association. Delete the instance using the intrinsic operation deleteInstance.

If this is the last instance of a CIM_ElementSettingData association for a particular CIM_ZoneMembershipSettingData, do not delete the instance of CIM_ZoneMembershipSettingData; it is the provider's responsibility to clean up these structures.

6.7.13 Intrinsic for deleting a zone member

Zone members are represented by CIM_ZoneMembershipSettingData instances associated to zones or zone aliases via CIM_ElementSettingData associations. To delete a zone member (and remove it from any zones or zone aliases from which it is a member) use the CIM operation deleteInstance to delete the instance of CIM_ZoneMembershipSettingData.

Do not delete the corresponding instances of the CIM_ElementSettingData; it is the provider's responsibility to clean up these structures.

6.7.14 Intrinsic for deleting a zone, zone alias, or zone set

Use the intrinsic operation deleteInstance to delete a zone, zone alias or zone set. Client are allowed to delete zones or zone aliases that are members of collections (zones or zone sets). Clients are allowed to delete the last member of a zone or zone set, leaving the collection empty.

A zone set or zone cannot be deleted if it is currently active (the error would be CIM_ERR_FAILED). Some implementations may prohibit deleting zonesets, zones or zone aliases that still have members (the error would be CIM_ERR_FAILED). When a zone, zone alias or zone set is deleted, the client does not have to delete the corresponding instances of CIM_MemberOfCollection or CIM_HostedCollection; it is the provider's responsibility to clean up these structures.

6.8 Client Considerations and Recipes

Many agent implementations do not allow Zone, a ZoneAlias or a Zone Set to be defined empty. Since the methods defined in SMI-S do not support creating a Zone Set with a Zone and a Zone with a Zone Member, the SessionControl method should be used to build a Zone Definition that is interoperable. This is done by calling ZoneSession() to “Start” defining or updating the Zone Definition. The client then calls the appropriate methods as necessary to build the desired Zone Definition. For example, calling CreateZoneSet() to create a new Zone Set, CreateZone() to create a new Zone, AddZoneToZoneSet() to add the newly created Zone to the newly created Zone Set, and CreateZoneMembershipSettingData() to create and add a new Zone Member to the newly created Zone. Upon completion of the new zoning definition, ZoneControl is called again to “End” the session. The changes to the Zone Definition would then be applied to the Zone Set Database. This set of calls would create a Zone Definition where the Zone and ZoneSet are not empty and would be interoperable across all agent implementations.

6.8.1 Create or delete zones Common Functions

```
// DESCRIPTION
//
// Common functions used by the recipes below.
//
// startSession: attempt to start fabric session if required;
//   returns false if attempt fails; returns true if attempt succeeds
//   or if session control is unnecessary
//
// endSession: finalize fabric session if required; returns false
//   if attempt fails; returns true if attempt succeeds or if session
//   control is unnecessary
//
//
// findActiveZoneSet: routine to find the active zoneset
```

```
// on a fabric, and return the reference to it
//


// PREEXISTING CONDITIONS AND ASSUMPTIONS
//
// None

sub boolean startSession ($ZoneService->)
{
    $ZoneService = GetInstance($ZoneService->, false, false, false, null)

    // session statuses
    #Ended = 3
    #NotApplicable = 4

    // requested session statuses
    #Started = 2
    #NoChange = 5

    if ($ZoneService.SessionState == #NotApplicable)
         return true // no session control implemented by this agent

    if ($ZoneService.SessionState != #Ended)
         return false // fabric session is in use by another client or agent

    if ($ZoneService.RequestedSessionState != #NoChange)
         return false // another client has already requested session

    %InArguments[“RequestedSessionState”] = #Started

    #status = InvokeMethod($ZoneService->, “SessionControl”, %InArguments,
                           %OutArguments)
    if (#status != 0) // e.g. “Failed”
        return false

    $ZoneService = GetInstance($ZoneService->, false, false, false, null)
    if ($ZoneService.SessionState != #Started)
        return false

    return true
}


// PREEXISTING CONDITIONS AND ASSUMPTIONS
//
// None
```

```
sub boolean endSession ($ZoneService->) {
    $ZoneService = GetInstance($ZoneService->, false, false, false, null)

    // session statuses
    #Started = 2
    #NotApplicable = 4

    // requested session statuses
    #End = 3

    if ($ZoneService.SessionStatus == #NotApplicable){
        return true      // no need for session control

    if ($ZoneService.SessionStatus != #Started)
        return false     // no session started by this client

    %InArguments[“RequestedSessionState”] = #End
    #status = InvokeMethod($ZoneService, “SessionControl”, %InArguments,
                           %OutArguments)
    if (#status != 0)   // e.g. “Failed”
        return false

    // Do not wait, or even check, for SessionState to have value “Ended” as
    // a) InvokeMethod will block till done (or failed) anyway
    // b) Before the check can be made, session may already be started
    //    by another client

    return true
}


// PREEXISTING CONDITIONS AND ASSUMPTIONS
// The reference to the fabric on which the active
// zoneset it to be sought is already known in
// the input variable $Fabric. Calling code
// should verify that the returned reference is non-null
//
sub Ref findActiveZoneSet($Fabric->){
     $ActiveZoneSet->=null
     $ZoneSets[] = Associators(
            $Fabric->,
            “CIM_HostedCollection”,
            “CIM_ZoneSet”,
            null,
            null,
            false,
            false,
            {“Active”} )
```

```
        // there may be no active zoneset
        if(0 < ZoneSets[].size()){
            for(#i in $ZoneSets[]){
                if(true==$ZoneSets[#i].Active){
                    $ActiveZoneSet->=nameof($ZoneSets[#i])
                    break
                }
            }
        }
        return $ActiveZoneSet->
}
```

6.8.2 Add new Zone Member to Existing Zone

```
// DESCRIPTION
// Add new Zone Member to Existing Zone
//
// Assume the client has already invoked some logic to determine which
// System (fabric or switch) will host the zone database and zone
// service to be used.  Request and obtain a fabric session from the
// zone service.  Use an extrinsic method to attempt to create a new
// instance of ZoneMembershipSettingData, associated to a zone.  If
// the creation fails because an instance already exists for the
// desired zone member id, simply create an association between the
// pre-existing ZoneMembershipSettingData instance and the zone
// instance.  Then close the fabric session.
//
// PREEXISTING CONDITIONS AND ASSUMPTIONS
//
// 1. The System hosting the zone database (ComputerSystem or
//    AdminDomain) has been previously identified and defined in the
//    $System-> variable
//
// 2. The zone member type is defined in the #ConnectivityMemberType variable
//
// 3. The zone member id of the new zone member is defined in the
//    #ConnectivityMemberID variable
//
// 4. An existing zone is defined in the $Zone-> variable
//
// FUNCTIONS

// 1. Get the Zone Service and start the session

$ZoneServices->[] = AssociatorNames($System->, "CIM_HostedService",
                                    "CIM_ZoneService", null, null)
```

```
// Assumption 1 (above) guarantees there is a zone service for this
// System, Fabric Profile mandates there is no more than one zone
// service for this System
$ZoneService-> = $ZoneService->[0]

// Start the session
if (!&startSession($ZoneService->)) {
    <ERROR! Failed to start zone session>
}

// 2. Create a ZoneMembershipSettingData
%InArguments["ConnectivityMemberType"] = #ConnectivityMemberType
%InArguments["ConnectivityMemberID"] = #ConnectivityMemberID
%InArguments["SystemSpecificCollection"] = $Zone->
#status = InvokeMethod($ZoneService->, "CreateZoneMembershipSettingData",
                       %InArguments[], %OutArguments[])

if (#status != 0){
     <ERROR! call to method CreateZoneMembershipSettingData failed #status>
}
// 3. Store the returned object path for verification
$ZoneMember-> = %OutArguments["ZoneMembershipSettingData"]

// 4. End session successfully
if(!&endSession($ZoneService->)){
     <ERROR! Failed to end session, changes may not have been committed>
}

// 5. Verify that the zonemember exist within the specified zone

$ZoneMembers->[]=associatorNames(
        $Zone->,
        "CIM_ElementSettingData",
        "CIM_ZoneMembershipSettingData",
        "ManagedElement",
        "SettingData" )
if(!contains($ZoneMember->,$ZoneMembers[])){
     <ERROR! Failed to verify zone member created>
}
```

6.8.3 Create new Zone, add new Zone Member, and add to existing ZoneSet

```
// DESCRIPTION
// Create new Zone, add new Zone Member, and add to existing ZoneSet
//
// Assume the client has already invoked some logic to determine which
// System (fabric or switch) will host the zone database and zone
```

```
// service to be used.  Request and obtain a fabric session from the
// zone service.  Create a new Zone using an extrinsic method.  The
// session may not be ended if any zone is empty, so add a zone member
// to the new zone.  The session also may not be ended unless every
// zone is a member of at least one zone set, so add the new zone to
// an existing zone set.  Then close the fabric session.
//
//
// PREEXISTING CONDITIONS AND ASSUMPTIONS
//
// 1. The System hosting the zone database (ComputerSystem or
//    AdminDomain) has been previously identified and defined in the
//    $System-> variable
//
// 2. The name for a new zone is defined in the #ZoneName variable
//
// 3. The type for the new zone is defined in the #ZoneType variable
//
// 4. The sub type for the new zone is defined in the #ZoneSubType
//    variable
//
// 5. The zone member type is defined in the #ConnectivityMemberType variable
//
// 6. The zone member id of the new zone member is defined in the
//    #ConnectivityMemberID variable
//
// 7. An existing zoneSet is defined in the $ZoneSet-> variable
//
// FUNCTIONS

// 1. Get the Zone Service and start the session
$ZoneServices->[] = AssociatorNames($System->, "CIM_HostedService",
                                    "CIM_ZoneService", null, null)

// Assumption 1 (above) guarantees there is a zone service for this
// System, Fabric Profile mandates there is no more than one zone
// service for this System
$ZoneService-> = $ZoneServices->[0]

    if (!&startSession($ZoneService->)) {
    <ERROR! Failed to start zone session>
}

// 2. Create a zone
%InArguments["ZoneName"] = #ZoneName
%InArguments["ZoneType"] = #ZoneType
%InArguments["ZoneSubType"] = #ZoneSubType
```

```
InvokeMethod($ZoneService->, "CreateZone", %InArguments[], %OutArguments[])
$Zone-> = $OutArguments["Zone"]

// 3. Create  a ZoneMembershipSettingData
%InArguments["ConnectivityMemberType"] = #ConnectivityMemberType
%InArguments["ConnectivityMemberID"] = #ConnectivityMemberID
%InArguments["SystemSpecificCollection"] = $Zone->
#status = InvokeMethod($ZoneService->, "CreateZoneMembershipSettingData",
                       %InArguments[], %OutArguments[])

if (#status != 0){
     <ERROR! Call to method CreateZoneMembershipSettingData failed #status>
}
// 4. Save the returned member objectpath for verification
$ZoneMember-> = %OutArguments["ZoneMembershipSettingData"]


// 5. Add the new zone to the existing zone set
%InArguments["ZoneSet"] = $ZoneSet->
%InArguments["Zone"] = $Zone->
#status = InvokeMethod($ZoneService->, "AddZone", %InArguments[], %OutArguments[])
if (#status != 0){
     <ERROR Call to method AddZone failed>
}
// 6. End Session
if(!&endSession($ZoneService->)){
     <ERROR! Failed to end session, changes may not have been committed>
}
// 7. Verify that the zone exists in the zone set
$Zones->[]=associatorNames(
        $ZoneSet->,
        "CIM_MemberOfCollection",
        "CIM_Zone",
        "Collection",
        "Member"
        )
// see if the zone is in the returned array
if(!contains($Zone->,$Zones->[])){
     <ERROR! Failed to verify that Zone was added to ZoneSet>
}
```

6.8.4 Create new ZoneSet and add existing Zone

```
// DESCRIPTION
// Create new ZoneSet and add existing Zone
```

```
//
// Assume the client has already invoked some logic to determine which
// System (fabric or switch) will host the zone database and zone
// service to be used.  Request and obtain a fabric session from the
// zone service.  Create a new ZoneSet with a given name, using an
// extrinsic method.  The session may not be ended if any ZoneSet is
// empty, so add an existing zone to the ZoneSet. Then close the
// fabric session.
//
// PREEXISTING CONDITIONS AND ASSUMPTIONS
//
// 1. The System hosting the zone database (ComputerSystem or
//    AdminDomain) has been previously identified and defined in the
//    $System-> variable
//
// 2. The name for the new zone set is defined in the #ZoneSetName
//    variable
//
// 3. An existing zone is defined in the $Zone-> variable
//
// FUNCTIONS

// 1. Get the Zone Service and start the session

$ZoneServices->[] = AssociatorNames($System->, "CIM_HostedService",
                                    "CIM_ZoneService", null, null)
// Assumption 1 (above) guarantees there is a zone service for this
// System, Fabric Profile mandates there is no more than one zone
// service for this System
$ZoneService-> = $ZoneServices->[0]

if (!&startSession($ZoneService->)){
     <ERROR! Failed to start zone session>
}

// 2. Create a zone set
%InArguments["ZoneSetName"] = #ZoneSetName
#status = InvokeMethod($ZoneService->, "CreateZoneSet", %InArguments[],
                           %OutArguments[])
if (#status != 0){
     <ERROR! Call to method CreateZoneSet failed>
}

$ZoneSet-> = %OutArguments["ZoneSet"]

// 3. Add the existing zone to the new zone set
%InArguments["ZoneSet"] = $ZoneSet->
%InArguments["Zone"] = $Zone->
```

```
#status = InvokeMethod($ZoneService->, "AddZone", %InArguments[], %OutArguments[])
if (#status != 0){
     <ERROR! Call to method AddZone failed #status>
}

// 4. End Session
if(!&endSession($ZoneService->)){
     <ERROR! Failed to end zone session, changes may not be committed>
}

// 5. Verify that the new zone set exists in the zone database
try{
     GetInstance($ZoneSet->);
}catch(CIM_ERR_NOT_FOUND){
     <ERROR! Failed to verify ZoneSet created>
}
```

6.8.5 Delete zone

```
// DESCRIPTION
// Delete Zone
//
// Try to use intrinsic delete operation to delete a Zone instance.
// Before any operations can be imposed on the zoning service, a
// session is requested and obtained from the zone service.  If the
// deletion fails, this may be because the zone is active, or because
// it is not empty.  In the latter case, remove all members from the
// zone by deleting the ElementSettingData association instances, and
// try the deletion again.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1.  The object name of the zone to be deleted is defined in the
//     $Zone-> variable
// 2.  The object name of the zone service object for the System
//     hosting the zone database is defined in the $ZoneService->
//     variable


if(!&startSession($ZoneService->)){
     <ERROR! Failed to start session>
}

try {
    DeleteInstance($Zone->)
}
catch(CIM_ERR_FAILED) {
    // Verify that Zone is not active
```

```
    $Zone = GetInstance($Zone->, false, false, false, null)
    if ($Zone.Active) {
        // tell client of its logic problem
        <ERROR! May not delete Zone from active ZoneSet>
    }

    // Failure may be caused because zone has members
    // Try to delete all zone memberships (not zone members themselves)
    $ZoneElements->[] = ReferenceNames($Zone->, "CIM_ElementSettingData", null)
    for #i in $ZoneElements->[] {
        DeleteInstance($ZoneElements[#i])
    }

    // Try again
    DeleteInstance($Zone->)
}
if(!&endSession($ZoneService->)){
     <ERROR! Failed to end session, changes may not be committed>
}
// Verify that the zone no longer exists in the zone database
try{
     GetInstance($Zone->)
}catch(CIM_ERR_NOT_FOUND){
     // expect failure
     return
}
// error if no exception thrown
<ERROR! Found Zone that should have been deleted>
```

6.8.6 Delete ZoneSet

```
// DESCRIPTION
// Delete Zone Set
//
// Try to use intrinsic delete operation to delete a ZoneSet
// instance. Before any operations can be imposed on the zoning
// service, a session is requested and obtained from the zone service.
// The session is released when the operations are complete.  If the
// deletion fails, this may be because the zone set is active, or
// because it is not empty.  In the latter case, remove all zones from
// the zone set by deleting the MemberOfCollection association
// instances, and try the deletion again.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1.  The object name of the zone set to be deleted is defined in the
//     $ZoneSet-> variable
```

```
// 2.  The object name of the zone service object for the system
//     hosting the zone database is defined in the $ZoneService->
//     variable

if (!&startSession($ZoneService->))
     <ERROR! Failed to start session>
}

try {
    DeleteInstance($ZoneSet->)
}
catch(CIM_ERR_FAILED) {
    $ZoneSet = GetInstance($ZoneSet->, false, false, false, null)
    if ($ZoneSet.Active) {
        // tell client of logic problem
        <ERROR! May not delete an active ZoneSet>
    }

    // Failure may be because zoneset is not empty
    $ZoneMemberships->[] = ReferenceNames($ZoneSet->, "CIM_MemberOfCollection",
                           null)
    for #i in $ZoneMemberships->[] {
        DeleteInstance($ZoneMemberships->[$i])
    }

    // Try again
    DeleteInstance($ZoneSet->)
}
if(!&endSession($ZoneService->)){
     <ERROR! Failed to end session, changes may not have been committed>
}
// Verify that the deletion did indeed occur
try{
     GetInstance($ZoneSet->)
}catch(CIM_ERR_NOT_FOUND){
     // expected, not a recipe error
     return
}
// error if no exception caught
<ERROR! Found ZoneSet that should have been deleted>
```

6.8.7 Delete ZoneMember

```
// DESCRIPTION
// Delete a zone member, removing it from any zones and aliases of
// which it is a member.
```

```
//
// Use the intrinsic delete operation to delete a
// ZoneMembershipSettingData instance. Before any operations can be
// imposed on the zoning service, a session is requested and obtained
// from the zone service.  The session is released when the operations
// are complete.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1.  The object name of the ZoneMembershipSettingData to be deleted is defined in
                          the
//     $ZoneMember-> variable
// 2.  The object name of the zone service object for the system
//     hosting the zone database is defined in the $ZoneService->
//     variable

if(!&startSession($ZoneService->)){
     <ERROR! Failed to start session>
}

DeleteInstance($ZoneMember->)
if(!&endSession($ZoneService->)){
     <ERROR! Failed to end session, changes may not have been committed>
}
// verify that it is indeed deleted
try{
     GetInstance($ZoneMember->)
}catch(CIM_ERR_NOT_FOUND){
     // expect an exception,
     // not a recipe error
     return
}
// error if no exception caught
<ERROR! Found ZoneMember that should have been deleted>
```

6.9 CIM Elements

Table 49 describes the CIM elements for Zone Control.

Table 49 - CIM Elements for Zone Control

Element Name	Requirement	Description
6.9.1 CIM_HostedService (Fabric (AdminDomain) to ZoneService)	Optional	Associates the ZoneService to the AdminDomain representing the fabric.

Table 49 - CIM Elements for Zone Control

Element Name	Requirement	Description
6.9.2 CIM_HostedService (Switch (ComputerSystem) to ZoneService)	Optional	Associates the ZoneService to the ComputerSystem representing the switch.
6.9.3 CIM_ZoneService (Zone Service)	Mandatory	The service that allows for all of the zoning configuration changes.

6.9.1 CIM_HostedService (Fabric (AdminDomain) to ZoneService)

Associates the ZoneService to the AdminDomain representing the fabric.

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

Table 50 describes class CIM_HostedService (Fabric (AdminDomain) to ZoneService).

Table 50 - SMI Referenced Properties/Methods for CIM_HostedService (Fabric (AdminDomain) to ZoneService)

Properties	Requirement	Description & Notes
Antecedent	Mandatory	The reference to the AdminDomain representing the fabric.
Dependent	Mandatory	The reference to the ZoneService.

6.9.2 CIM_HostedService (Switch (ComputerSystem) to ZoneService)

Associates the ZoneService to the ComputerSystem representing the switch.

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

Table 51 describes class CIM_HostedService (Switch (ComputerSystem) to ZoneService).

Table 51 - SMI Referenced Properties/Methods for CIM_HostedService (Switch (ComputerSystem) to ZoneService)

Properties	Requirement	Description & Notes
Antecedent	Mandatory	The reference to the ComputerSystem representing the switch.
Dependent	Mandatory	The reference to the ZoneService.

6.9.3 CIM_ZoneService (Zone Service)

The service that allows for all of the zoning configuration changes.

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

Table 52 describes class CIM_ZoneService (Zone Service).

Table 52 - SMI Referenced Properties/Methods for CIM_ZoneService (Zone Service)

Properties	Requirement	Description & Notes
SystemCreationClassName	Mandatory	The scoping System's CreationClassName.
SystemName	Mandatory	The scoping System's Name.
CreationClassName	Mandatory	The Class Name
Name	Mandatory	Opaque
OperationalStatus	Mandatory	Status of Zoning Service.
SessionState	Mandatory	State of session. Valid values are "Starting", "Ended".
RequestedSessionState	Mandatory	The requested session state from the client. The valid values that can be set are "Start", "End", and "Terminate".
DefaultZoningState	Mandatory	
CreateZoneSet()	Mandatory	The method creates a ZoneSet and associates it to the System (AdminDomain representing the Fabric or the ComputerSystem representing the Switch) that the ZoneService is hosted on.
CreateZone()	Mandatory	The method creates a Zone and associates it to System (AdminDomain representing the Fabric or the ComputerSystem representing the Switch) that the ZoneService is hosted on.
CreateZoneMembershipSettingData()	Mandatory	The method creates a ZoneMembershipSettingData (a zone member) and adds it to the specified Zone or NamedAddressCollection representing a Fibre Channel Node.
AddZone()	Mandatory	The method adds to the specified ZoneSet the specified Zone.
AddZoneMembershipSettingData()	Mandatory	The method adds to the specified Zone or NamedAddessCollection representing the Fibre Channel Node the specified ZoneMembershipSettingData (a zone member).
ActivateZoneSet()	Mandatory	The method activates the specified ZoneSet.
SessionControl()	Mandatory	The method enables a client to request a lock of the fabric to begin zoning configuration changes.

STABLE

IMPLEMENTED

Clause 7: FDMI Subprofile

7.1 Description

The Fabric-Device Management Interface (FDMI) enables the management of devices such as HBAs through the Fabric. The FDMI complements data in the Fabric Profile. It allows for any entity in the Fabric to expose through SMI the HBA information without having an agent resident on the Host containing the HBA.

This profile only addresses HBA type devices. The HBA Management Interface defined by FDMI is a subset of interface defined by the Fibre Channel HBA API specification, as exposed by *Storage Management Technical Specification, Part 6 Host Elements, 1.3.0 Rev 6* Clause 5: FC HBA Profile.

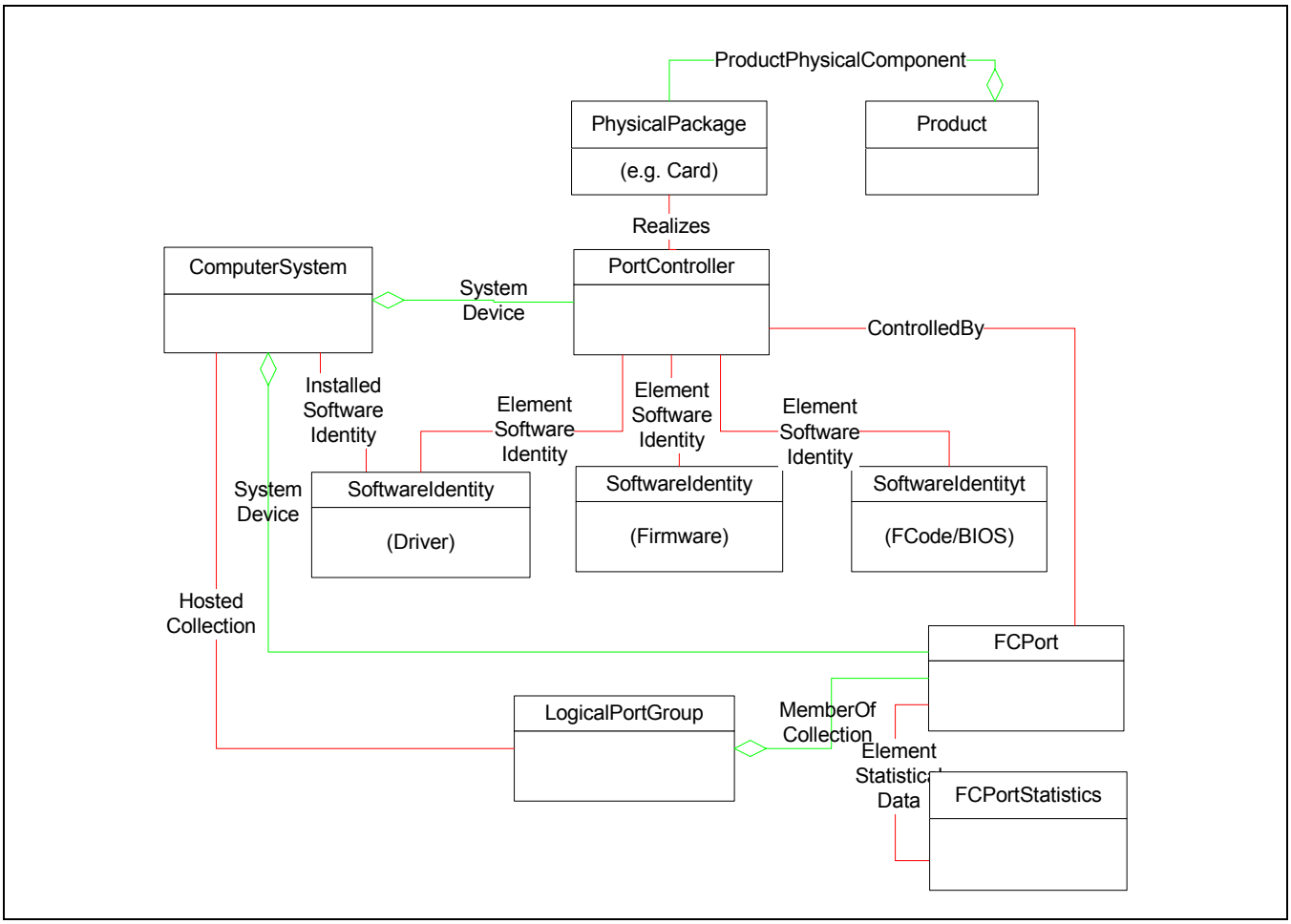

Figure 8 - FDMI Instance

7.2 Health and Fault Management

None

7.3 Cascading Considerations

None

7.4 Dependencies on Profiles, Subprofiles, and Packages

None

7.5 Methods of this Profile

None

7.6 Client Considerations and Recipes

None

7.7 Registered Name and Version

FDMI version 1.2.0

7.8 CIM Elements

Table 53 describes the CIM elements for FDMI.

Table 53 - CIM Elements for FDMI

Element Name	Requirement	Description
7.8.1 CIM_ComputerSystem	Mandatory	The System the HBA is within.
7.8.2 CIM_ControlledBy	Mandatory	Associates the ComputerSystem with the PortController
7.8.3 CIM_ElementSoftwareIdentity	Mandatory	Associates the SoftwareIdentity to the HBA
7.8.4 CIM_FCPort	Mandatory	The HBA Fibre Channel Port
7.8.5 CIM_HostedCollection	Mandatory	Associates the LogicalPortGroup (Fibre Channel Node) to the hosting System.
7.8.6 CIM_InstalledSoftwareIdentity	Mandatory	Associates the SoftwareIdentity representing the driver to the System it is installed on.
7.8.7 CIM_LogicalPortGroup	Mandatory	The Fibre Channel Node
7.8.8 CIM_MemberOfCollection	Mandatory	Associates FCPort to the LogicalPortGroup
7.8.9 CIM_PhysicalPackage	Mandatory	The physical package that the HBA is contained in
7.8.10 CIM_PortController	Mandatory	The HBA
7.8.11 CIM_Product	Mandatory	The product information for the HBA
7.8.12 CIM_ProductPhysicalComponent	Mandatory	Associates the Product to the PhysicalPackage
7.8.13 CIM_Realizes	Conditional	Conditional requirement: C2Associates the PhysicalPackage to the PortController
7.8.14 CIM_SoftwareIdentity (Driver)	Mandatory	The software for the driver

Table 53 - CIM Elements for FDMI

Element Name	Requirement	Description
7.8.15 CIM_SoftwareIdentity (Firmware)	Mandatory	The software for the firmware
7.8.16 CIM_SoftwareIdentity (Option ROM)	Mandatory	The software for the Option ROM
7.8.17 CIM_SystemDevice	Mandatory	Associates the ComputerSystem with the FCPort
7.8.18 CIM_SystemDevice (ComputerSystem to FCPort)	Optional	Associates the FCPort to the ComputerSystem
7.8.19 CIM_SystemDevice (ComputerSystem to PortController)	Mandatory	Associates the ComputerSystem with the PortController
7.8.20 CIM_SystemDevice (Switch to FCPort)	Optional	Associates the FCPort to the ComputerSystem
7.8.21 CIM_SystemDevice (System to FCPort)	Optional	Associates the FCPort to the ComputerSystem

7.8.1 CIM_ComputerSystem

The system the HBA is within. It is identified using Host Name from the FDMI interface.

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

Table 54 describes class CIM_ComputerSystem.

Table 54 - SMI Referenced Properties/Methods for CIM_ComputerSystem

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Name		Mandatory	The name of the host containing the Device. The key identifier helping in discovery to determine which HBAs are in the same host.
NameFormat		Mandatory	

7.8.2 CIM_ControlledBy

Associates the ComputerSystem with the PortController.

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

Table 55 describes class CIM_ControlledBy.

Table 55 - SMI Referenced Properties/Methods for CIM_ControlledBy

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

7.8.3 CIM_ElementSoftwareIdentity

Associates the SoftwareIdentities representing the various software for the HBA to the PortController representing the HBA.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 56 describes class CIM_ElementSoftwareIdentity.

Table 56 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

7.8.4 CIM_FCPort

The HBA Fibre Channel Port.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 57 describes class CIM_FCPort.

Table 57 - SMI Referenced Properties/Methods for CIM_FCPort

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	The scoping System's CreationClassName.
SystemName		Mandatory	The scoping System's Name.
CreationClassName		Mandatory	Name of Class
DeviceID		Mandatory	Opaque

Table 57 - SMI Referenced Properties/Methods for CIM_FCPort

Properties	Flags	Requirement	Description & Notes
ElementName		Mandatory	Port Symbolic Name if available. Otherwise NULL. If the underlying implementation includes characters that are illegal in CIM strings, then truncate before the first of those characters.
LinkTechnology		Mandatory	'FC'
PermanentAddress		Mandatory	Fibre Channel Port WWN
NetworkAddresses	C	Mandatory	Fibre Channel ID (FCID). Expressed as 8 un-separated upper case hex digits (see Table 4 for more information about formats).
ActiveFC4Types		Mandatory	The active Fibre Channel FC-4 protocol
SupportedFC4Types		Optional	An array of integers indicating the Fibre Channel FC-4 protocols supported **EXPERIMENTAL** When supporting FC-SB, this property shall be either '27' (FC-SB-2 Channel) for Host Ports or '28' (FC-SB-2 Control Unit) for targets.
SupportedCOS		Optional	An array of integers indicating the Fibre Channel Classes of Service that are supported.
PortType		Mandatory	The specific port type currently enabled (from FC-GS Port.Type)
Speed		Optional	Speed of zero represents a link not established. 1Gb is 1062500000 bps 2Gb is 2125000000 bps 4Gb is 4250000000 bps 10Gb single channel variants are 10518750000 bps 10Gb four channel variants are 12750000000 bps This is the raw bit rate.

7.8.5 CIM_HostedCollection

Associates the LogicalPortGroup (Fibre Channel Node) to the hosting System.

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

Table 58 describes class CIM_HostedCollection.

Table 58 - SMI Referenced Properties/Methods for CIM_HostedCollection

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

7.8.6 CIM_InstalledSoftwareIdentity

Associates the SoftwareIdentity representing the driver to the System it is installed on.

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

Table 59 describes class CIM_InstalledSoftwareIdentity.

Table 59 - SMI Referenced Properties/Methods for CIM_InstalledSoftwareIdentity

Properties	Flags	Requirement	Description & Notes
InstalledSoftware		Mandatory	
System		Mandatory	

7.8.7 CIM_LogicalPortGroup

The Fibre Channel Node

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

Table 60 describes class CIM_LogicalPortGroup.

Table 60 - SMI Referenced Properties/Methods for CIM_LogicalPortGroup

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
Name	D	Mandatory	Fibre Channel Node WWN
NameFormat		Mandatory	'WWN'
ElementName	N	Mandatory	Node Symbolic Name if available. Otherwise NULL. If the underlying implementation includes characters that are illegal in CIM strings, then truncate before the first of those characters.

7.8.8 CIM_MemberOfCollection

Associates FCPort to the LogicalPortGroup

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

Table 61 describes class CIM_MemberOfCollection.

Table 61 - SMI Referenced Properties/Methods for CIM_MemberOfCollection

Properties	Flags	Requirement	Description & Notes
Collection		Mandatory	
Member		Mandatory	

7.8.9 CIM_PhysicalPackage

The physical package that the HBA is contained by. It can be simply a PhysicalPackage that the system and HBA is contained within. If it is known that the HBA is on a separate board, Card (a subclass of PhysicalPackage) can be used.

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

Table 62 describes class CIM_PhysicalPackage.

Table 62 - SMI Referenced Properties/Methods for CIM_PhysicalPackage

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	Name of Class
Tag		Mandatory	An arbitrary string that uniquely identifies the PhysicalPackage.
ElementName		Optional	User Friendly name. This property is OPTIONAL.
Name		Optional	
Manufacturer		Mandatory	
Model		Mandatory	
SerialNumber		Optional	
Version		Optional	
PartNumber		Optional	

7.8.10 CIM_PortController

The HBA. The HBA may have logical operations that can apply to it (e.g. OperationalStatus).

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

Table 63 describes class CIM_PortController.

Table 63 - SMI Referenced Properties/Methods for CIM_PortController

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
DeviceID		Mandatory	
ControllerType		Mandatory	

7.8.11 CIM_Product

The product information for the HBA

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

Table 64 describes class CIM_Product.

Table 64 - SMI Referenced Properties/Methods for CIM_Product

Properties	Flags	Requirement	Description & Notes
Name		Mandatory	Commonly used Product name.
IdentifyingNumber		Mandatory	Product identification such as a serial number.
Vendor		Mandatory	The manufacturer or the OEM.
Version		Mandatory	Product version information.
ElementName		Mandatory	User Friendly name. Suggested use is Vendor, Version and product name.

7.8.12 CIM_ProductPhysicalComponent

Associates the Product to the PhysicalPackage. This is necessary to link the Product information to the HBA.

Created By: Static

Modified By: Static
Deleted By: Static
Requirement: Mandatory

Table 65 describes class CIM_ProductPhysicalComponent.

Table 65 - SMI Referenced Properties/Methods for CIM_ProductPhysicalComponent

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

7.8.13 CIM_Realizes

Associates the PhysicalPackage to the PortController.

Created By: Static
Modified By: Static
Deleted By: Static
Requirement: null

Table 66 describes class CIM_Realizes.

Table 66 - SMI Referenced Properties/Methods for CIM_Realizes

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	
Antecedent		Mandatory	
Dependent		Mandatory	

7.8.14 CIM_SoftwareIdentity (Driver)

The software for the driver

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

Table 67 describes class CIM_SoftwareIdentity (Driver).

Table 67 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity (Driver)

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
VersionString		Mandatory	

Table 67 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity (Driver)

Properties	Flags	Requirement	Description & Notes
Manufacturer		Mandatory	
BuildNumber		Optional	
MajorVersion		Optional	
RevisionNumber		Optional	
MinorVersion		Optional	
Classifications		Mandatory	

7.8.15 CIM_SoftwareIdentity (Firmware)

The software for the firmware

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

Table 68 describes class CIM_SoftwareIdentity (Firmware).

Table 68 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity (Firmware)

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
VersionString		Mandatory	
Manufacturer		Mandatory	
BuildNumber		Optional	
MajorVersion		Optional	
RevisionNumber		Optional	
MinorVersion		Optional	
Classifications		Mandatory	

7.8.16 CIM_SoftwareIdentity (Option ROM)

The software for the Option ROM

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

Table 69 describes class CIM_SoftwareIdentity (Option ROM).

Table 69 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity (Option ROM)

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
VersionString		Mandatory	
Manufacturer		Mandatory	
BuildNumber		Optional	
MajorVersion		Optional	
RevisionNumber		Optional	
MinorVersion		Optional	
Classifications		Mandatory	

7.8.17 CIM_SystemDevice

Associates the ComputerSystem with the FCPort

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

Table 70 describes class CIM_SystemDevice.

Table 70 - SMI Referenced Properties/Methods for CIM_SystemDevice

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

7.8.18 CIM_SystemDevice (ComputerSystem to FCPort)

Associates the FCPort to the ComputerSystem

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

Table 71 describes class CIM_SystemDevice (ComputerSystem to FCPort).

Table 71 - SMI Referenced Properties/Methods for CIM_SystemDevice (ComputerSystem to FCPort)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

7.8.19 CIM_SystemDevice (ComputerSystem to PortController)

Associates the ComputerSystem with the PortController

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

Table 72 describes class CIM_SystemDevice (ComputerSystem to PortController).

Table 72 - SMI Referenced Properties/Methods for CIM_SystemDevice (ComputerSystem to Port-Controller)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

7.8.20 CIM_SystemDevice (Switch to FCPort)

Associates the FCPort to the ComputerSystem

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

Table 73 describes class CIM_SystemDevice (Switch to FCPort).

Table 73 - SMI Referenced Properties/Methods for CIM_SystemDevice (Switch to FCPort)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

7.8.21 CIM_SystemDevice (System to FCPort)

Associates the FCPort to the ComputerSystem

Created By: Static

Modified By: Static
Deleted By: Static
Requirement: Optional

Table 74 describes class CIM_SystemDevice (System to FCPort).

Table 74 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to FCPort)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

IMPLEMENTED

EXPERIMENTAL

Clause 8: Fabric Path Performance Subprofile

8.1 Description

The fabric path performance subprofile extends the standard capabilities of obtaining performance associated to a port to identify the performance in the path defined by an initiator and target ProtocolEndpoint. In the current networking model, the path through the “cloud” is defined by NetworkPipe which is a class that is associated to a ProtocolEndpoint by EndpointOfNetworkPipe. Since the statistics model is defined to allow an association to any LogicalElement, the statistics collected for an NetworkPort, NetworkPortStatistics, can be associated to the NetworkPipe also. When a device supports the Fabric Path Performance Subprofile, it will instantiate the NetworkPipe and as it collects statistics will instantiate the StatisticalData.

The class, StatisticsCollection, provides a mechanism to “collect” all the statistics associated to the NetworkPipes.

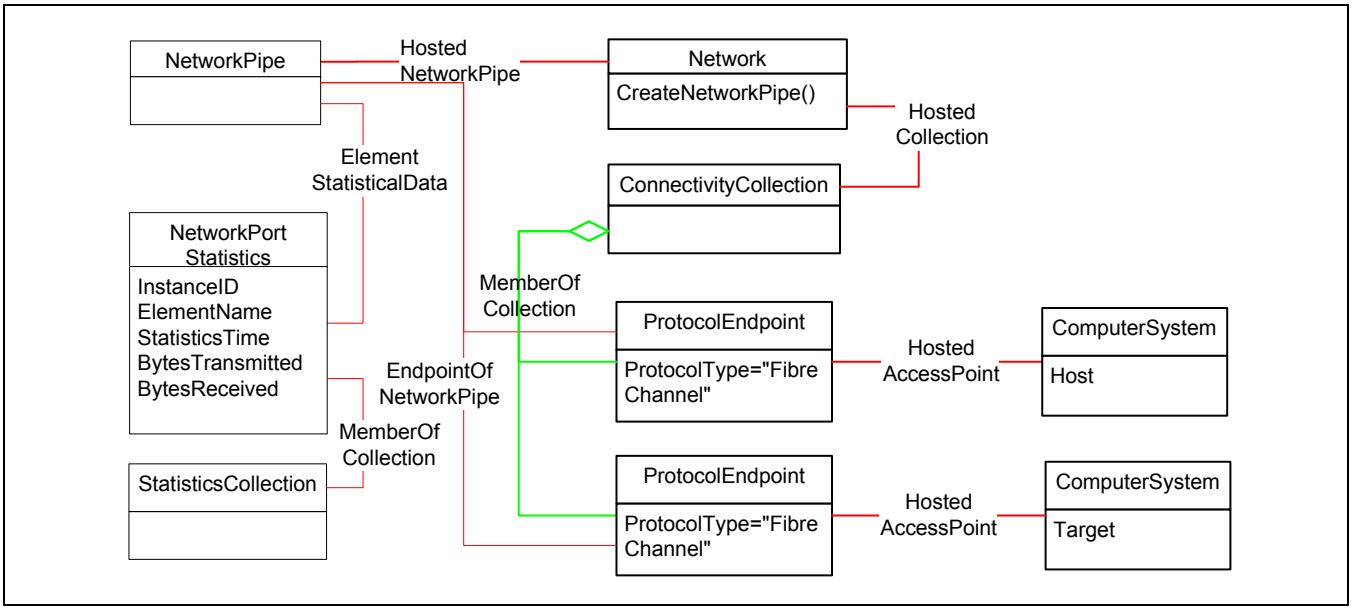

Figure 9 - Instance Diagram

8.2 Health and Fault Management

None

8.3 Dependencies on Profiles, Subprofiles, and Packages

None

8.4 Methods of this Profile

The method CreateNetworkPipes is used to indicate to the underlying instrumentation that statistics shall be collected between the sets of ServiceAccessPoints. A NetworkPipe maybe created for each set of statistics needed, though the underlying hardware may have maximum limitations. If the creation of the NetworkPipes are successful, the client can then discover the NetworkPipes and also the associated statics. It is implementation specific how long the statistics are collected. For instance, the NetworkPipes may persist as long as the hardware is not rebooted.

```
CreateNetworkPipes (
        [IN] Directionality[],
        [IN] CIM_ServiceAccessPoint REF SourceSAP[],
        [IN] CIM_ServiceAccessPoint REF SinkSAP[],
        [OUT] CIM_ConcreteJob REF Job,
        [OUT] CIM_NetworkPipe REF NetworkPipe[]);
```

8.5 Client Considerations and Recipes

None

8.6 Registered Name and Version

Fabric Path Performance version 1.2.0

8.7 CIM Elements

Table 75 describes the CIM elements for Fabric Path Performance.

Table 75 - CIM Elements for Fabric Path Performance

Element Name	Requirement	Description
8.7.1 CIM_ElementStatisticalData	Mandatory	Associates FCPortStatistics to the FCPort
8.7.2 CIM_EndpointOfNetworkPipe	Mandatory	Associates NetworkPipe to the ProtocolEndpoint
8.7.3 CIM_HostedCollection	Mandatory	Associates the Statistics Collection to the Network representing the fabric.
8.7.4 CIM_HostedNetworkPipe	Mandatory	Associates NetworkPipe to the Network
8.7.5 CIM_MemberOfCollection	Mandatory	Associates the NetworkPortStatistics to the StatisticsCollection.
8.7.6 CIM_Network	Mandatory	Subclass of AdminDomain representing the fabric
8.7.7 CIM_NetworkPipe	Mandatory	Pipe through the cloud from an initiator to the target.
8.7.8 CIM_NetworkPortStatistics	Mandatory	NetworkPort Statistics of the NetworkPipe
8.7.9 CIM_ProtocolEndpoint	Mandatory	The initiator or target (ends of the NetworkPipe).
8.7.10 CIM_StatisticsCollection	Mandatory	Collection to aggregate the NetworkPipe statistics

8.7.1 CIM_ElementStatisticalData

Associates the NetworkPortStatistics being collected by the monitor to the NetworkPipe (representing the monitor and defining the path being monitored)

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

Table 76 describes class CIM_ElementStatisticalData.

Table 76 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	Reference to NetworkPipe representing the monitor that collects the statistics
Stats		Mandatory	Reference to NetworkPortStatistics that the monitor is collecting

8.7.2 CIM_EndpointOfNetworkPipe

Associates NetworkPipe representing the monitor to the ProtocolEndpoint. The set of associations (both ends of the NetworkPipe) represent the path that statistics are collected for.

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

Table 77 describes class CIM_EndpointOfNetworkPipe.

Table 77 - SMI Referenced Properties/Methods for CIM_EndpointOfNetworkPipe

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	Reference to ProtocolEndpoint representing one end of the path to collect statistics for
Dependent		Mandatory	Reference to NetworkPipe representing the monitor

8.7.3 CIM_HostedCollection

Associates the Statistics Collection to the Network representing the fabric.

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

Table 78 describes class CIM_HostedCollection.

Table 78 - SMI Referenced Properties/Methods for CIM_HostedCollection

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	Reference to ConnectivityCollection which collects all ProtocolEndpoints in a fabric
Antecedent		Mandatory	Reference to Network representing the fabric

8.7.4 CIM_HostedNetworkPipe

Associates NetworkPipe to the Network representing the fabric. To find all the monitors running in a fabric, this association should be used.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 79 describes class CIM_HostedNetworkPipe.

Table 79 - SMI Referenced Properties/Methods for CIM_HostedNetworkPipe

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	Reference to Network representing fabric
Dependent		Mandatory	Reference to NetworkPipe representing the monitor that collects the statistics

8.7.5 CIM_MemberOfCollection

Associates the NetworkPortStatistics to the StatisticsCollection which collects all of the statistics for fabric path performance

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 80 describes class CIM_MemberOfCollection.

Table 80 - SMI Referenced Properties/Methods for CIM_MemberOfCollection

Properties	Flags	Requirement	Description & Notes
Collection		Mandatory	Reference to StatisticsCollection that collects all statistics that are for fabric path performance
Member		Mandatory	Reference to the statistics for fabric path performance

8.7.6 CIM_Network

Subclass of AdminDomain that represents the fabric. NetworkPipes are scoped to the Network via the association HostedNetworkPipe. The Network has the method CreateNetworkPipe() to create the NetworkPipes needed to define monitors within the fabric to collect the path statistics.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 81 describes class CIM_Network.

Table 81 - SMI Referenced Properties/Methods for CIM_Network

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	Name of class.
Name		Mandatory	WWN of Fabric
NameFormat		Mandatory	"WWN"
NameFormat		Mandatory	
CreateNetworkPipe()		Optional	

8.7.7 CIM_NetworkPipe

The NetworkPipe for this profile is instantiated to provide a mechanism to indicate monitors are in place in the network to collect statistical information. NetworkPortStatistics are associated to the pipe via the association ElementStatisticalData to NetworkPortStatistics and subclasses of NetworkPortStatistics (e.g. FCPortStatistics).

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 82 describes class CIM_NetworkPipe.

Table 82 - SMI Referenced Properties/Methods for CIM_NetworkPipe

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
ElementName		Optional	

8.7.8 CIM_NetworkPortStatistics

Network Port Statistics represent a snapshots of counters for the NetworkPipe. An instance of this class can represent the statistics for the current statistics, archived and consolidated statistics, or both.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 83 describes class CIM_NetworkPortStatistics.

Table 83 - SMI Referenced Properties/Methods for CIM_NetworkPortStatistics

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Optional	TBS
StatisticTime		Optional	The time the statistics were collected.
BytesTransmitted		Mandatory	The total number of bytes that are transmitted, including framing characters.
BytesReceived		Mandatory	The total number of bytes that are received, including framing characters.

8.7.9 CIM_ProtocolEndpoint

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

Table 84 describes class CIM_ProtocolEndpoint.

Table 84 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	
NameFormat		Mandatory	
ProtocolIFType		Mandatory	

8.7.10 CIM_StatisticsCollection

The collection, hosted to the Network representing the Fabric, is used to aggregate all the statistics for the NetworkPipes.

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

Table 85 describes class CIM_StatisticsCollection.

Table 85 - SMI Referenced Properties/Methods for CIM_StatisticsCollection

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	TBS
SampleInterval		Mandatory	The interval statistics the statistics associated to this collection are collected. This value indicates to clients the minimum period statistics should be sampled.
TimeLastSampled		Mandatory	TBS

EXPERIMENTAL

EXPERIMENTAL

Clause 9: Fibre Channel Security Subprofile

9.1 Description

Fibre Channel Security Subprofile describes the Fibre Channel Security Protocol (FC-SP) management policies. This profile is a specialization of the Authorization Subprofile.

The FC-SP Policies can be broken down into four parts:

- Membership
- Connectivity
- Management Access
- Attributes

FC-SP Membership policy is composed of two components, Switch Membership and Device Membership. Switch Membership is fabric-wide information which defines which switches are allowed to be part of a fabric, controls physical management access, and allows switch characteristics to be specified. Device Membership is fabric-wide information which defines which devices are allowed to be part of a fabric, controls in-band management access, and allows device characteristics to be specified. FC-SP Connectivity policy defines connectivity restrictions on a per-switch basis.

This subprofile describes exposing the FC-SP Membership and Connectivity policies. It does not define physical management and switch characteristics as part of the Membership policy. The specialization of the Authorization Subprofile primarily incudes the subclassing of Identity to StorageHarwareID to identify the AuthorizedSubject and AuthorizedTargets.

For the Membership Policy, the AuthrorizedTarget is always the Fabric identified by its Fabric Name and the two possible AuthorizedSubjects are the switch identified by its switch name (WWN) and a device by its node WWN. In Figure 10, the diagram shows both a switch having membership and a device having membership.

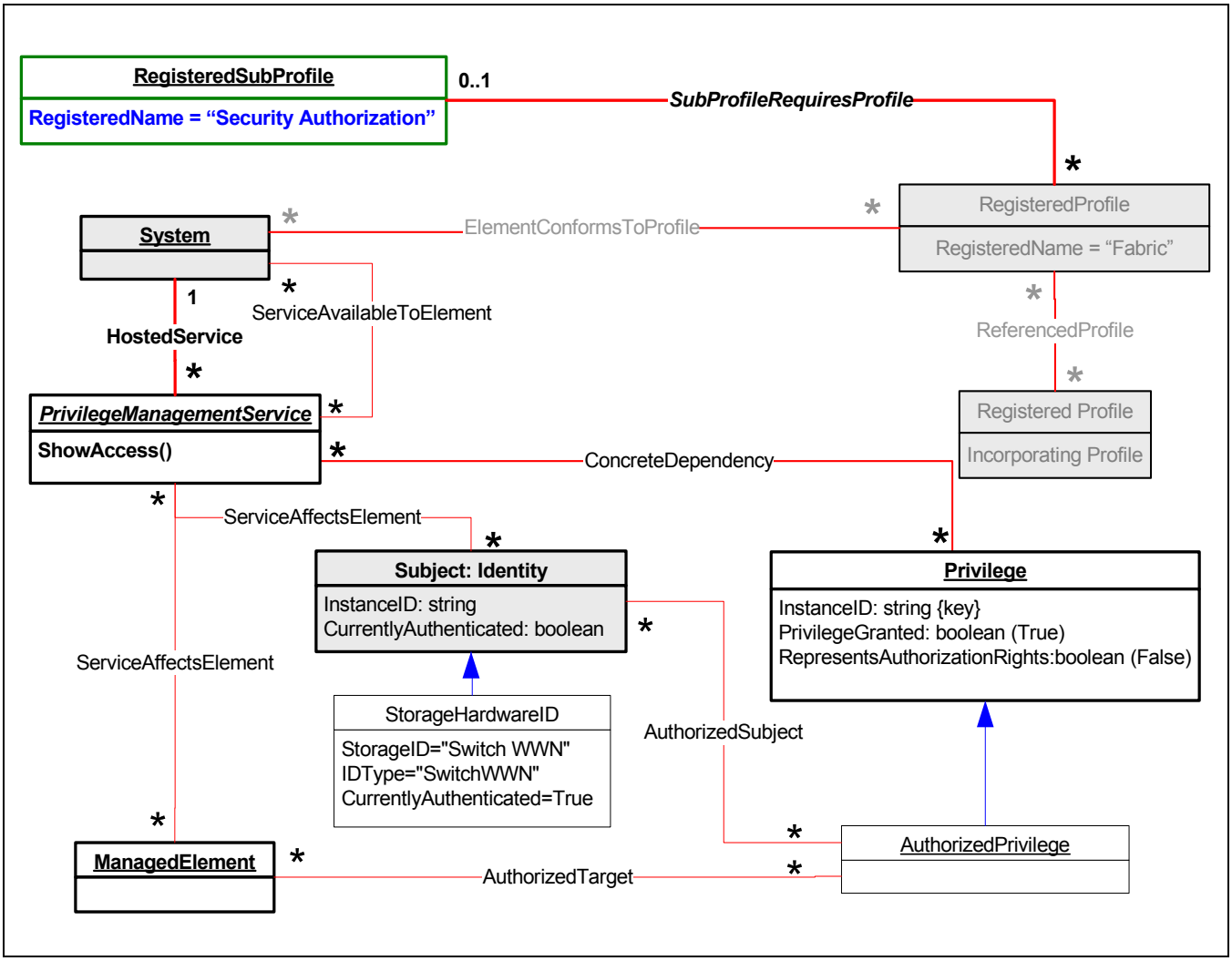

Figure 10 - Specialization of Security Authorization Subprofile for Membership Policy

Figure 11 shows both a switch having Switch Membership and a device having Device Membership. A switch is identified by its switch name (WWN) and a device by its node WWN.

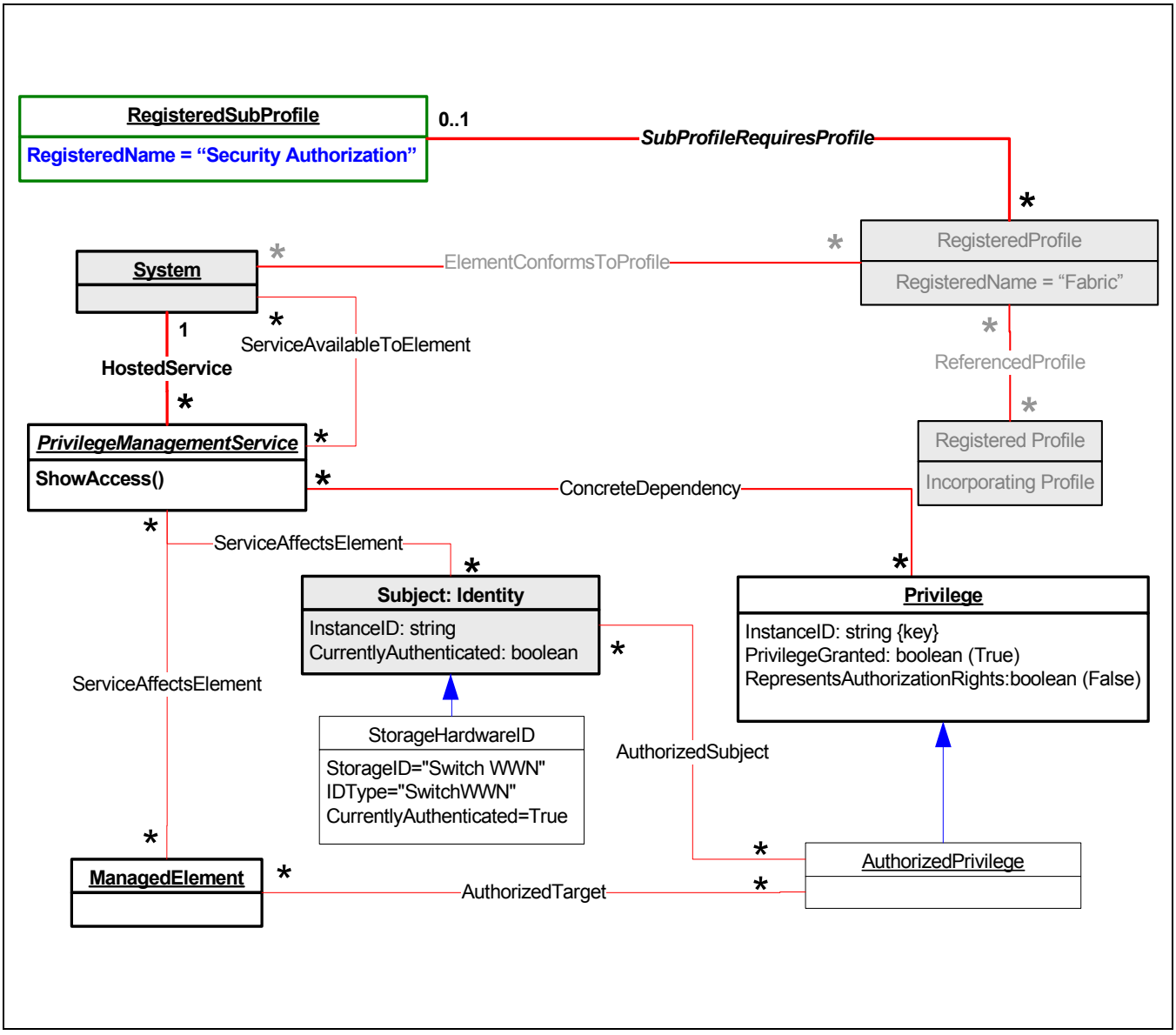

Figure 11 - Specialization of Security Authorization Subprofile for Connectivity Policy

9.2 Health and Fault Management Consideration

None

9.3 Cascading Considerations

None

9.4 Supported Profiles, Subprofiles, and Packages

None

9.5 Methods of the Profile

9.6 Client Considerations and Recipes

None

9.7 Registered Name and Version

FabricSecurity version 1.2.0

9.8 CIM Elements

Table 86 describes the CIM elements for FabricSecurity.

Table 86 - CIM Elements for FabricSecurity

Element Name	Requirement	Description
9.8.1 CIM_AuthorizationService	Mandatory	The service controling the security policy in the fabric
9.8.2 CIM_AuthorizedPrivilege	Mandatory	Privilege granted to the Switch or the Node
9.8.3 CIM_AuthorizedSubject	Mandatory	The association of the Node or Switch to its Privilege.
9.8.4 CIM_AuthorizedTarget	Mandatory	The association of the privilege (or a switch or node) to the target (switch or the fabric) that the switch or node is being granted access to.
9.8.5 CIM_HostedService	Mandatory	Associates the AuthorizationService to the AdminDomain that is hosting it
9.8.6 CIM_ServiceAffectsElement (ManagedElement to Service)	Mandatory	Service Affects Managed Element
9.8.7 CIM_ServiceAffectsElement (StorageHardwareID to Service)	Mandatory	Service affects StorageHardwareID
9.8.8 CIM_ServiceAvailableToElement (Fabric AdminDomain to Service)	Mandatory	Service available to fabric
9.8.9 CIM_StorageHardwareID	Mandatory	The class that identifies the subject to be granted access

9.8.1 CIM_AuthorizationService

Created By: Static
Modified By: Static

Deleted By: Static
Requirement: Mandatory

Table 87 describes class CIM_AuthorizationService.

Table 87 - SMI Referenced Properties/Methods for CIM_AuthorizationService

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	Name of System class
SystemName		Mandatory	Name of class
CreationClassName		Mandatory	Name of AuthorizationService class
Name		Mandatory	Uniquely identifies the service in the fabric

9.8.2 CIM_AuthorizedPrivilege

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

Table 88 describes class CIM_AuthorizedPrivilege.

Table 88 - SMI Referenced Properties/Methods for CIM_AuthorizedPrivilege

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
PrivilegeGranted		Mandatory	
RepresentsAuthorizationRights		Mandatory	

9.8.3 CIM_AuthorizedSubject

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

Table 89 describes class CIM_AuthorizedSubject.

Table 89 - SMI Referenced Properties/Methods for CIM_AuthorizedSubject

Properties	Flags	Requirement	Description & Notes
PrivilegedElement		Mandatory	
Privilege		Mandatory	

9.8.4 CIM_AuthorizedTarget

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

Table 90 describes class CIM_AuthorizedTarget.

Table 90 - SMI Referenced Properties/Methods for CIM_AuthorizedTarget

Properties	Flags	Requirement	Description & Notes
Privilege		Mandatory	
TargetElement		Mandatory	Reference to target element

9.8.5 CIM_HostedService

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

Table 91 describes class CIM_HostedService.

Table 91 - SMI Referenced Properties/Methods for CIM_HostedService

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	AdminDomain representing the Fabric
Dependent		Mandatory	AuthorizationService

9.8.6 CIM_ServiceAffectsElement (ManagedElement to Service)

Created By: Static
Modified By: Static
Deleted By: Static

Requirement: Mandatory

Table 92 describes class CIM_ServiceAffectsElement (ManagedElement to Service).

Table 92 - SMI Referenced Properties/Methods for CIM_ServiceAffectsElement (ManagedElement to Service)

Properties	Flags	Requirement	Description & Notes
AffectingElement		Mandatory	AuthorizationService
AffectedElement		Mandatory	AdminDomain representing the Switch

9.8.7 CIM_ServiceAffectsElement (StorageHardwareID to Service)

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

Table 93 describes class CIM_ServiceAffectsElement (StorageHardwareID to Service).

Table 93 - SMI Referenced Properties/Methods for CIM_ServiceAffectsElement (StorageHardwareID to Service)

Properties	Flags	Requirement	Description & Notes
AffectingElement		Mandatory	AuthorizationService
AffectedElement		Mandatory	AdminDomain representing the Switch

9.8.8 CIM_ServiceAvailableToElement (Fabric AdminDomain to Service)

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

Table 94 describes class CIM_ServiceAvailableToElement (Fabric AdminDomain to Service).

Table 94 - SMI Referenced Properties/Methods for CIM_ServiceAvailableToElement (Fabric AdminDomain to Service)

Properties	Flags	Requirement	Description & Notes
ServiceProvided		Mandatory	AuthorizationService
UserOfService		Mandatory	AdminDomain representing the Fabric

9.8.9 CIM_StorageHardwareID

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

Table 95 describes class CIM_StorageHardwareID.

Table 95 - SMI Referenced Properties/Methods for CIM_StorageHardwareID

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
StorageID		Mandatory	Node WWN, Switch WWN, Port WWN
IDType		Mandatory	"NodeWWN" "SwitchWWN""or PortWWN"
CurrentlyAuthenticated		Mandatory	

EXPERIMENTAL

EXPERIMENTAL

Clause 10: Fabric Views Subprofile

10.1 Description

10.1.1 Synopsis

10.1.2 Overview

10.1.3 Overview

This Profile specifies SNIA_ View Classes for the Fabric Profiles.

In this release of SMI-S, SNIA_ view classes provide an optimization of retrieval of information provided by multiple (associated) instances in a Profile. There is no support for update of SNIA_ view classes instances. Update of a SNIA_ view class instance can only be accomplished by updating the base class instances from which the view is derived.

10.1.3.1 Goals of SNIA_ View Classes

10.1.3.1.1 Goals that SNIA_ View Classes are intended to address are:

- Get more data in one call to CIM Server.

  The CIM model for fabrics involve a number of classes and associations. The objective is to allow discovery of the fabric model using SNIA_ View Classes with a reduction in the number of association traversals required.

- Allow providers to optimize the Request.

  In many cases, the data represented by a View Class is actually kept (and returned) by a device as one entity. When the "normalized" CIM model is traversed many calls are made to retrieve that one entity. The provider takes the data from the one entity and carves it up for each CIM request. In many cases this involves retrieving the same entity multiple times. The objective is to allow a Provider to return the single entity in one SMI-S request (for data that is typically kept together by the device).

10.1.3.1.2 Additional Goals

- Do more things in one call to CIM Server.

  An example would be retrieval or discovery of model information with fewer calls. However, this goal also extends to updating the CIM model (e.g., configuration actions). The SNIA_ View Classes are NOT intended to help in the latter case. However, SNIA_ View Classes should facilitate access to underlying classes in support of configuration operations.

  It is important to note that the SNIA_ View Classes proposal was based directly on experiences relating to the scalability and performance of SMI-S real-world implementations. The focus is on improving performance in large configurations (e.g. thousands of volumes and thousands of disk drives).

10.1.3.2 Specific Requirements and Objectives of View Classes

10.1.3.2.1 Pre-defined View Classes

In order to gain the desired performance advantage, it is felt that view classes would have to be pre-defined (in SMI-S) to allow provider optimization of the requested information.

- Enable Associator Calls to View Class instances.

It should be possible to retrieve a View Class by an associators call to the class.

However, it is desired that the association should be clearly distinguished from existing associations on the base classes.

- Enable Associator Calls from View Class instances.

  It should be possible to get related classes (e.g., base classes) from the View Class by using associator calls.

  Again, the associations used should be clearly distinguished from existing associations on the base classes.

- Allow View Classes to be used where real classes would

  This certainly includes "read" intrinsics and as parameters of Extrinsics

  However, at this time "Write" intrinsic support is deferred and use in Extrinsics (as IN or OUT parameters) is not covered in this release of SMI-S.

10.1.3.2.2 Support Life Cycle Indications on View Classes

This requirement is being deferred for consideration in a future release of SMI-S.

10.1.4 Topology and Switch Views

Discovering a topology for a large fibre channel fabric (thousands of ports) time-wise is expensive. The existing fabric profile requires traversals of three associations and two classes for each link. Using the view class reduces this to enumeration of the view class itself for each link. The SNIA_TopologyView, shown in Figure 12, provides the properties to be able to build a topology without having to do associators from the ProtocolEndpoint to the FCPort to the FCNode or FCSwitch.

Discovering a fibre channel switch and its associated ports is also expensive. The SNIA_SwitchView, shown in Figure 12, provides the properties to be able to discover all the switch properties for a Fibre Channel Switch and Ports without having to do associators from the ComputerSystem representing the Switch to FCPort, LogicalModeule, Product, PhysicalPackage, SoftwareIdentity, AccessPoint, and Location.")

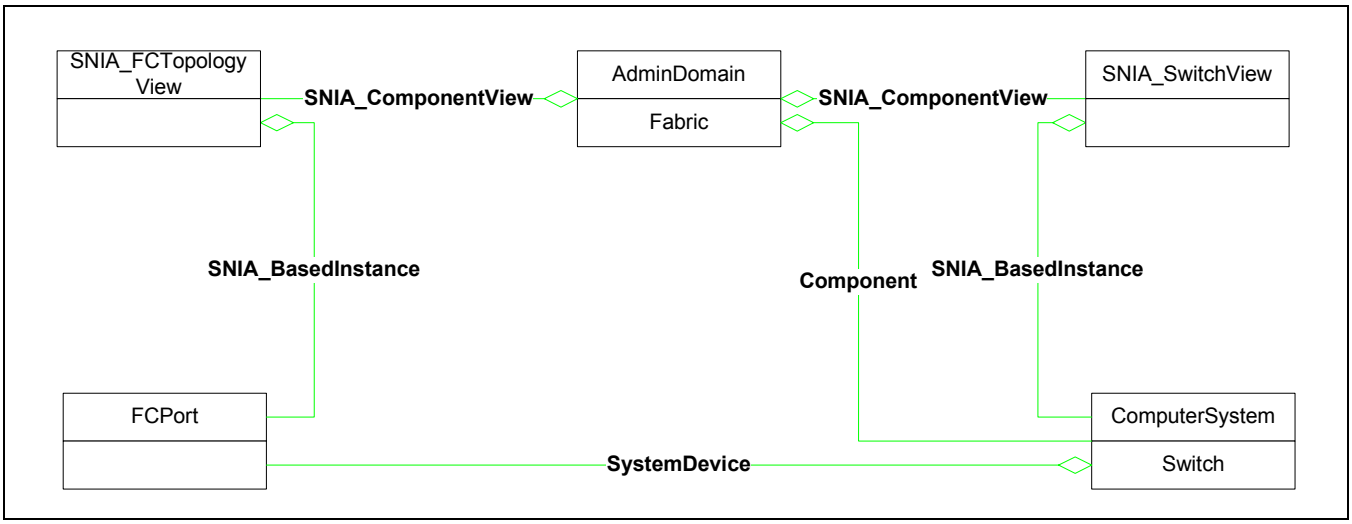

Figure 12 - Topology and Switch View Class

10.2 Health and Fault Management Consideration

None

10.3 Cascading Considerations

None

10.4 Supported Profiles, Subprofiles, and Packages

None

10.5 Methods of the Profile

None

10.6 Client Considerations and Recipes

None

10.7 CIM Element

10.8 Registered Name and Version

Fabric Views version 1.2.0

10.9 CIM Elements

Table 96 describes the CIM elements for Fabric Views.

Table 96 - CIM Elements for Fabric Views

Element Name	Requirement	Description
10.9.1 SNIA_FCSwitchView	Mandatory	The SNIA_SwitchView provides the properties for a Fibre Channel Switch.
10.9.2 SNIA_TopologyView	Mandatory	The SNIA_TopologyView provides the properties to be able to build a topology for a Fibre Channel Fabric.

10.9.1 SNIA_FCSwitchView

The SNIA_SwitchView provides the properties to be able to discover all the switch properties for a Fibre Channel Switch and Ports without having to do associators from the ComputerSystem representing the Switch to FCPort, LogicalModule, Product, PhysicalPackage, SoftwareIdentity, AccessPoint, and Location.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 97 describes class SNIA_FCSwitchView.

Table 97 - SMI Referenced Properties/Methods for SNIA_FCSwitchView

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
SwitchWWN	C	Mandatory	A Name_Identifier that is associated with a Fibre Channel switch.
DomainID		Mandatory	The Domain ID of the Switch.
ElementName		Optional	The Symbolic Name of the Switch.
OperationalStatus		Mandatory	The operational status of the Switch.
SwitchEnabledState		Mandatory	EnabledState indicates the enabled and disabled states of the Fibre Channel Switch.
SwitchNPIVSupported		Mandatory	Flag indicating whether NPIV is support on the fibre channel switch.
FCPortWWN	C	Mandatory	The FC Port Name Identifier (FC Port WWN).
FCPortElementName		Mandatory	The user friendly name of the FCPort.
FCPortType		Mandatory	The PortType of the FCPort.)
FCPortOperationalStatus		Mandatory	The operational status of the FCPort.
FCPortEnabledState		Mandatory	EnabledState indicates the enabled and disabled states of the Fibre Channel Port.
PortNumber		Mandatory	The port number labeled for an administrative purposes relative to the logical module. This is typically what is exposed through the devices element manager or CLI.
LogicalModulePortNumber		Mandatory	The port number labeled for an administrative purposes relative to the logical module.
DomainPortNumber		Mandatory	The port number used for domain id and physical port zone members.
ModuleNumber		Mandatory	Logical modules are often named by the physical or logical slot that they occupy within the Switch.
PortNPIVSupported		Mandatory	Flag indicating whether NPIV is support on the fibre channel port.
MaxPortNPIVLogins		Mandatory	Maximum number of NPIV logins supported on the fibre channel port.
Vendor		Mandatory	The name of the Product's supplier of the Switch.
ProductName		Mandatory	Commonly used Product name.
VersionString		Mandatory	A string representing the complete software version of the Switch.
Model		Mandatory	

Table 97 - SMI Referenced Properties/Methods for SNIA_FCSwitchView

Properties	Flags	Requirement	Description & Notes
SerialNumber		Mandatory	
PartNumber		Mandatory	
OEM		Mandatory	
OEMProductName		Mandatory	
OEMIdentifyingNumber		Mandatory	
AccessInfo		Mandatory	
LocationName		Mandatory	
LocationPhysicalPosition		Mandatory	

10.9.2 SNIA_TopologyView

The SNIA_TopologyView provides the properties to be able to build a topology for a Fibre Channel fabric without having to do associators from the ProtocolEndpoint to the FCPort to the FCNode or FCSwitch.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 98 describes class SNIA_TopologyView.

Table 98 - SMI Referenced Properties/Methods for SNIA_TopologyView

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
AntecedentFCPortWWN	C	Mandatory	The FC Port Name Identifier (FC Port WWN) of the FCPort the antecedent reference is associated to.
AntecedentElementWWN	C	Mandatory	The FC Switch Name Identifier (FC Switch WWN) or FC Node Name (FC Node WNN) the AntecedentFCPort is associated to.
AntecedentFCPortElementName		Mandatory	The user friendly name of the FCPort the antecedent reference is associated to.
AntecedentFCPortType		Mandatory	The PortType of the FCPort the antecedent reference is associated to.)
DependentFCPortWWN	C	Mandatory	The FC Port Name Identifier (FC Port WWN) of the FCPort the Dependent reference is associated to.
DependentElementWWN	C	Mandatory	The FC Switch Name Identifier (FC Switch WWN) or FC Node Name (FC Node WNN) the DependentFCPort is associated to.

Table 98 - SMI Referenced Properties/Methods for SNIA_TopologyView

Properties	Flags	Requirement	Description & Notes
DependentFCPortElementName		Mandatory	The user friendly name of the FCPort the dependent reference is associated to.
DependentFCPortType		Mandatory	The PortType of the FCPort the dependent reference is associated to.)
AntecedentFCPort		Mandatory	The antecedent FCPort that is communicating with the dependent FCPort.
AntecedentSystem		Mandatory	System of the antecedent FCPort in the AntecedentFCPort SystemDevice association.
DependentFCPort		Mandatory	The dependent FCPort that is communicating with the antecedent FCPort.
DependentSystem		Mandatory	System of the Dependent FCPort in the DependentFCPort SystemDevice association.

EXPERIMENTAL

EXPERIMENTAL

Clause 11: Virtual Fabrics Subprofile

11.1 Description

Technologies have recently been implemented for Fibre Channel SANs to logically separate the hardware into multiple fabrics yet keep them physically interconnected. The term for this technology defined by ANSI T11 is "Virtual Fabrics". ANSI T11 calls the underlying hardware “Core Switches”. To be consistent with a more broader use model and DMTF schematics, this subprofile will call these the partitioning systems. ANSI T11 calls the switching construct that resides in the partitioning system, a virtual switch. Again for the broader use model and consistency with the DMTF, this system will be call the partitioned system. The Fabric Profile already provides the discovery of the “virtual fabrics” and the “virtual switches”. This subprofile allows for the discovery of the underlying partitioning system. The Switch Partitioning subprofile provides the mechanism for configuring the partitioning system.

All Virtual Fabrics shall be associated to the same SAN for the same underlying “physical” topology. Also the Registered Subprofile associates to the SAN object to allow the client to arrive at the proper SAN object without traversing upwards from the fabric instance. In Figure 13, the relationship is shown. Note also that the partitioning ComputerSystems are associated with the SAN.

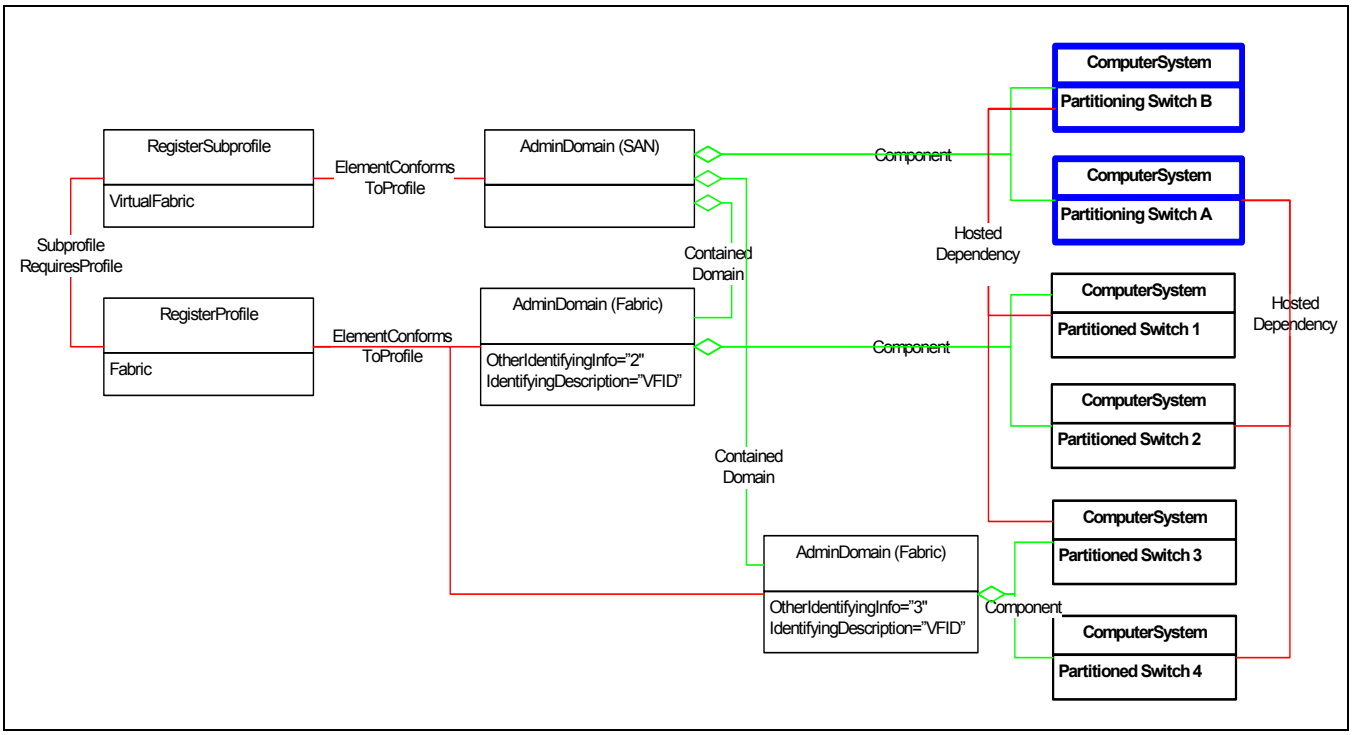

Figure 13 - RegisteredProfile/Subprofile, AdminDomain, and ComputerSystem Relationships

In Figure 14, the partitioning systems are shown presenting two “virtual fabric” with only one link (ActiveConnection) in each fabric. See Clause 4: Fabric Profile for more information with regards to fabric topology

using ActiveConnection. In this configuration each fabric, virtual switch, and port are uniquely identified. Note that the new instances to support this subprofile are outlined in a bolded line.

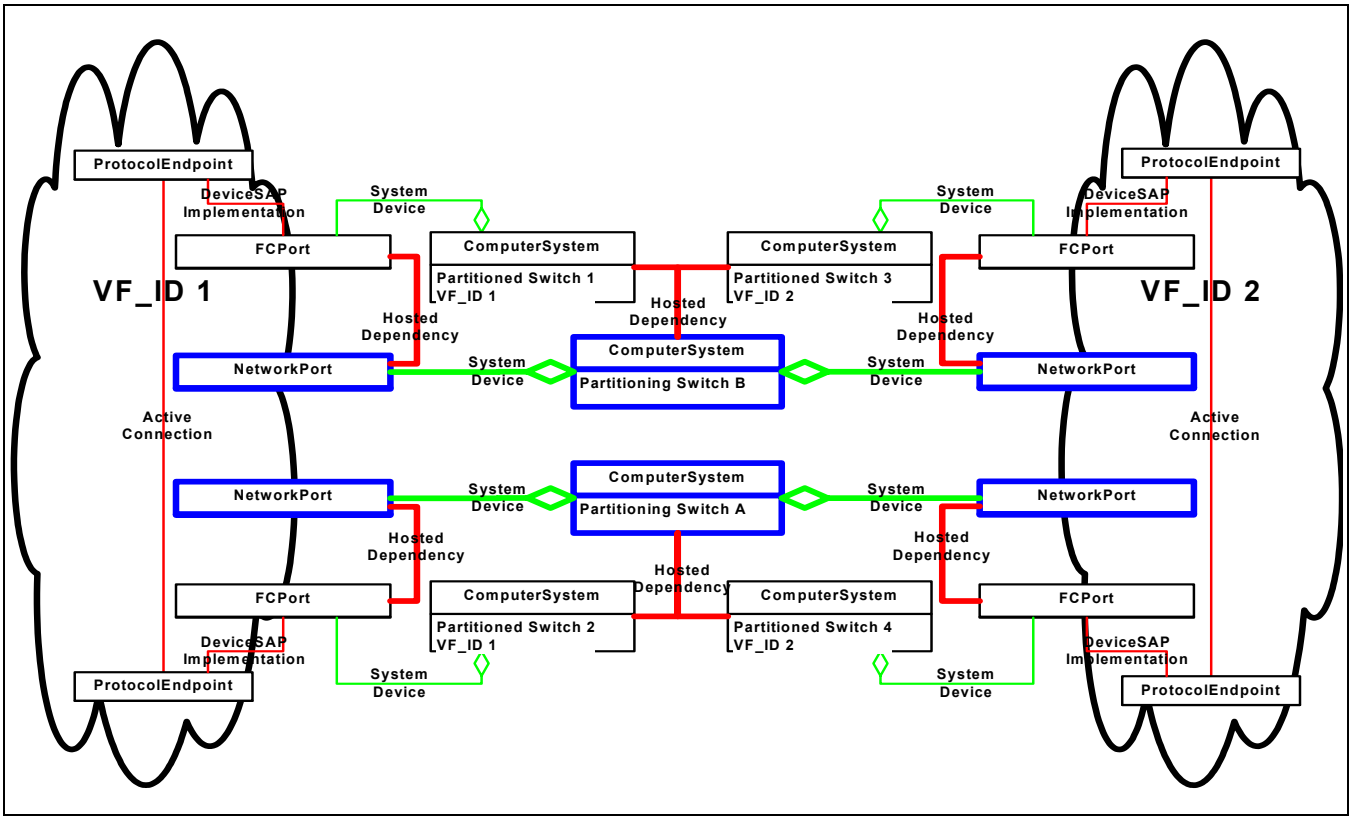

Figure 14 - Two Virtual Fabric and Two Partitioning Systems

In Figure 14, a single partitioning system is creating two virtual fabrics and in Figure 15, the system is partitioning

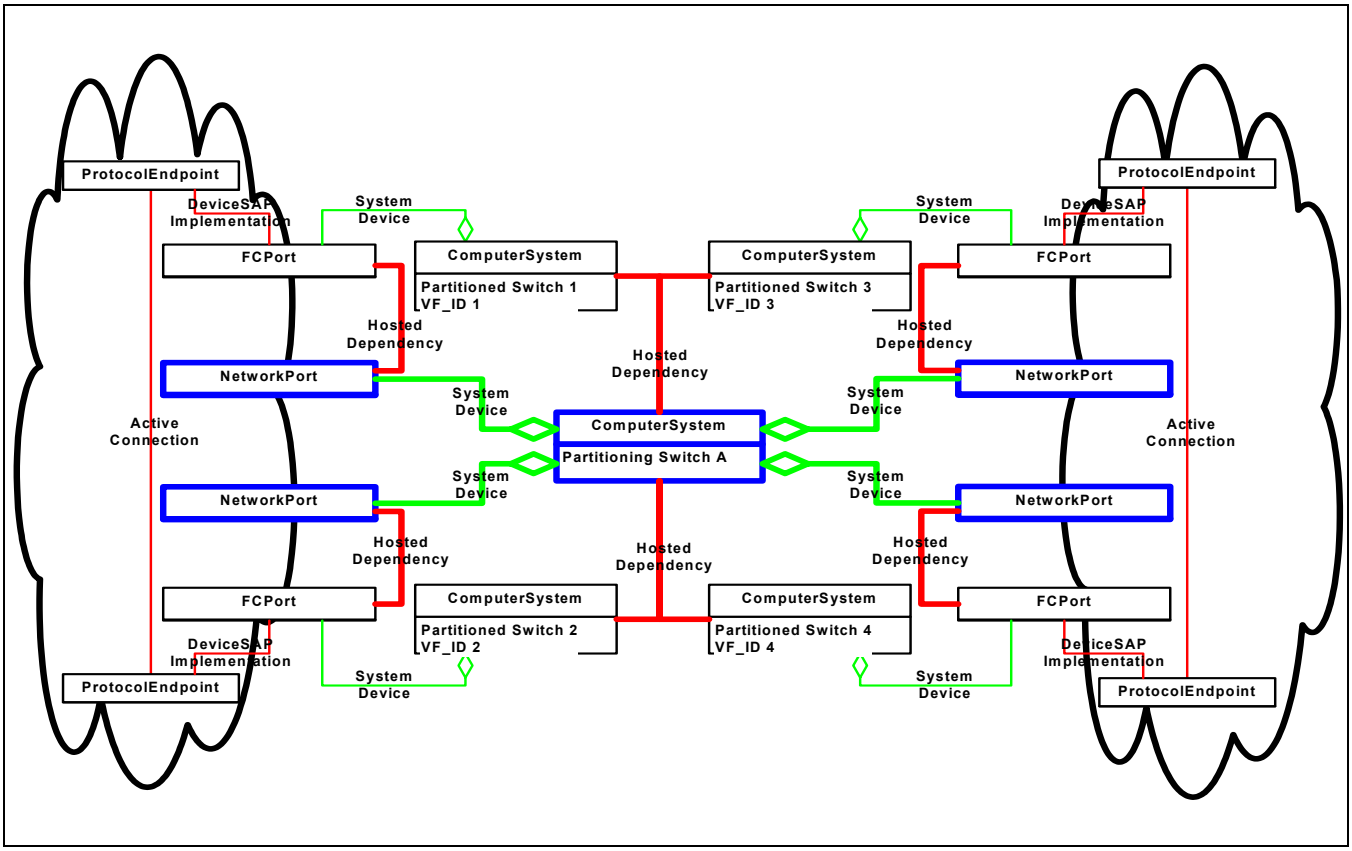

Figure 15 - Two Virtual Fabrics and One Partitioning System

the network port. When the NetworkPort is partitioned, each port is not guaranteed to be uniquely identified (though the key to the instance is) and the client needs to map it to the fabric it is participating in to maintain the ports identity.

In all cases you can have a one to one or many to one relationships between partitioned switches and the partitioning switch. The same is true between the partitioned FCPorts and the NetworkPorts.

To determine the end to end path of a fabric, the client follows the rules outlined in the fabric profile with the exception that when entering a switch, if there exists a hosted dependency on the switch port, the partitioning system shall be traversed instead of the partitioned system. So in Figure 15, when the client traverses across ActiveConnection to the ProtocolEndpoint to the FCPort and down the HostedDependency, it realizes that both ends of the link are actually in the same partitioned system.

11.2 Health and Fault Management Consideration

The possible Health and Fault information through LifeCycle indications are addressed in the Fabric Profile.

11.3 Cascading Considerations

None

11.4 Supported Profiles, Subprofiles, and Packages

Related Profiles for FabricVirtualFabrics: Not defined in this standard.

11.5 Methods of the Profile

None

11.6 Client Considerations and Recipes

None

11.7 Registered Name and Version

FabricVirtualFabrics version 1.2.0

11.8 CIM Elements

Table 99 describes the CIM elements for FabricVirtualFabrics.

Table 99 - CIM Elements for FabricVirtualFabrics

Element Name	Requirement	Description
11.8.1 CIM_AdminDomain (Fabric)	Mandatory	AdminDomain representing the Fabric.
11.8.2 CIM_AdminDomain (SAN)	Mandatory	AdminDomain representing the SAN
11.8.3 CIM_Component (AdminDoman to Partitioning CS)	Mandatory	Associates the partitioning ComputerSystems representing the underlying physical switches to the AdminDomain representing the SAN
11.8.4 CIM_ComputerSystem (Partitioned)	Mandatory	The partitioned ComputerSystem.
11.8.5 CIM_ComputerSystem (Partitioning)	Mandatory	The partitioning ComputerSystem.
11.8.6 CIM_HostedDependency (NetworkPort to FCPort)	Mandatory	Association between NetworkPort to FCPort.
11.8.7 CIM_HostedDependency (Partitioning CS to Partitioned CS)	Mandatory	Association between the Partitioning ComputerSystem and Partitioned ComputerSystem.
11.8.8 CIM_NetworkPort	Mandatory	The partitioning port.
11.8.9 CIM_SystemDevice (NetworkPort to ComputerSystem)	Mandatory	Associates the partitioning classes (NetworkPort to the ComputerSystem)

11.8.1 CIM_AdminDomain (Fabric)

AdminDomain representing the fabric. This is a logical entity and can represent virtual fabrics.

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

Table 100 describes class CIM_AdminDomain (Fabric).

Table 100 - SMI Referenced Properties/Methods for CIM_AdminDomain (Fabric)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	Name of Class
Name	C	Mandatory	WWN of Fabric
NameFormat		Mandatory	'WWN'
OtherIdentifyingInfo	C	Mandatory	Virtual Fabric ID
IdentifyingDescriptions		Mandatory	'VF_ID' is placed into corresponding index of OtherIdentifyingInfo

11.8.2 CIM_AdminDomain (SAN)

AdminDomain representing the SAN

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

Table 101 describes class CIM_AdminDomain (SAN).

Table 101 - SMI Referenced Properties/Methods for CIM_AdminDomain (SAN)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	Name of Class
Name		Mandatory	An arbitrary name (implementation dependent)
NameFormat		Mandatory	Dependent on the arbitrary name chosen.

11.8.3 CIM_Component (AdminDoman to Partitioning CS)

Associates the partitioning ComputerSystems representing the underlying physical switches to the AdminDomain representing the SAN.This allows the client another means to determine which ComputerSystems are part of the Fabric versus those that are part of the underlying physical topology. ComputerSystems representing Fibre Channel switches are associated to the AdminDomain representing the fabric (see the Fabric Profile).

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

Table 102 describes class CIM_Component (AdminDoman to Partitioning CS).

Table 102 - SMI Referenced Properties/Methods for CIM_Component (AdminDoman to Partitioning CS)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	The reference to the AdminDomain representing the SAN
PartComponent		Mandatory	The reference to the partitioning ComputerSystem

11.8.4 CIM_ComputerSystem (Partitioned)

The ComputerSystem representing the Interconnect Element (e.g. a switch) or Platform (e.g. Host and Array). This System will always be the Dependent in the association HostedDependency. Traversing this association will place you at the System that is being partitioned.

Created By: External

Modified By: Static

Deleted By: External

Requirement: Mandatory

Table 103 describes class CIM_ComputerSystem (Partitioned).

Table 103 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Partitioned)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Name		Mandatory	
Dedicated		Mandatory	
OtherIdentifyingInfo	C	Mandatory	Virtual Fabric ID
IdentifyingDescriptions		Mandatory	'VF_ID' is placed into corresponding index of OtherIdentifyingInfo

11.8.5 CIM_ComputerSystem (Partitioning)

The ComputerSystem that "hosts" the partitioned ComputerSystems.

Created By: External

Modified By: Static

Deleted By: External

Requirement: Mandatory

Table 104 describes class CIM_ComputerSystem (Partitioning).

Table 104 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Partitioning)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Name		Mandatory	
Dedicated		Mandatory	

11.8.6 CIM_HostedDependency (NetworkPort to FCPort)

Association between NetworkPort to FCPort

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

Table 105 describes class CIM_HostedDependency (NetworkPort to FCPort).

Table 105 - SMI Referenced Properties/Methods for CIM_HostedDependency (NetworkPort to FCPort)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	NetworkPort representing the Partitioning System
Dependent		Mandatory	CFCPort representing the Partitioned System

11.8.7 CIM_HostedDependency (Partitioning CS to Partitioned CS)

Association between the Partitioning ComputerSystem and Partitioned ComputerSystem

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

Table 106 describes class CIM_HostedDependency (Partitioning CS to Partitioned CS).

Table 106 - SMI Referenced Properties/Methods for CIM_HostedDependency (Partitioning CS to Partitioned CS)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	ComputerSystem representing the Partitioning System
Dependent		Mandatory	ComputerSystem representing the Partitioned System

11.8.8 CIM_NetworkPort

The partitioning port.

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

Table 107 describes class CIM_NetworkPort.

Table 107 - SMI Referenced Properties/Methods for CIM_NetworkPort

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	The scoping System's CreationClassName.
SystemName		Mandatory	The scoping System's Name.
CreationClassName		Mandatory	Name of Class
DeviceID		Mandatory	Opaque

11.8.9 CIM_SystemDevice (NetworkPort to ComputerSystem)

Associates the NetworkPort to the ComputerSystem

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

Table 108 describes class CIM_SystemDevice (NetworkPort to ComputerSystem).

Table 108 - SMI Referenced Properties/Methods for CIM_SystemDevice (NetworkPort to ComputerSystem)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	The reference to the partitioning ComputerSystem
PartComponent		Mandatory	The reference to the NetworkPort

EXPERIMENTAL

STABLE

Clause 12: Switch Profile

12.1 Description

The switch profile models logical and physical aspects of a Fibre Channel switch. The ComputerSystem class constitutes the center of the switch model (and is the top level object which the profile registration points to). An instance of a ComputerSystem is identified as a switch by the property Dedicated set to “switch”.

This profile includes discovery components including ports, port statistics, product information, software, and chassis information. It also includes configuration of the switch including switch and port state change, port speed, switch name, symbolic names, and DomainID.

Both the Switch and Port have a capabilities class, FCSwitchCapabilities and FCPortCapabilities, respectively, defining which configuration options are supported by the switch. The capabilities define what components are configurable and any restrictions that apply. Except for state change, an associated settings class is defined for both the switch and port, FCSwitchSettings and FCPortSettings, which the client uses to request configuration changes to the Switch or Port, respectively. A setting does not necessarily result in a change to the underlying Switch or Port. The client can determine whether the setting was applied by looking at the associated property in the Switch or Port class.

The model for configuration is made up of three components, capabilities, settings, and the ManagedElements, ComputerSystem and FCPort. The capabilities define what components are configurable and any restrictions that apply, the settings define what the client requests, and the ManagedElements expose the actual changes that were applied.

The ComputerSystem (Dedicated as Switch) and FCPort classes have the method RequestStateChange() for requesting that the state be changed and an associated property RequestedState on the classes which indicates the current state change that has been requested. To determine whether the state change has completed, the property EnabledState can be examined to determine whether the device has completed the state change.

If a switch is modular, for instance if the switch is comprised of multiple blades on a backplane, LogicalModule can optionally be used to model each sub-module, and as an aggregation point for the switch ports. This is described in the Blade Subprofile.

FCPort describes the logical aspects of the port link and the data layers. PhysicalConnector models the physical aspects of a port. An instance of the FCPortStatistics class is expected for each instance of the FCPort class. FCPortStatistics expose real time port health and traffic information.

If the instrumentation is embedded in a switch, it shall provide a switch profile implementation for the hosting switch, and it may proxy a switch profile implementation for other switches reported in the Fabric Profile.

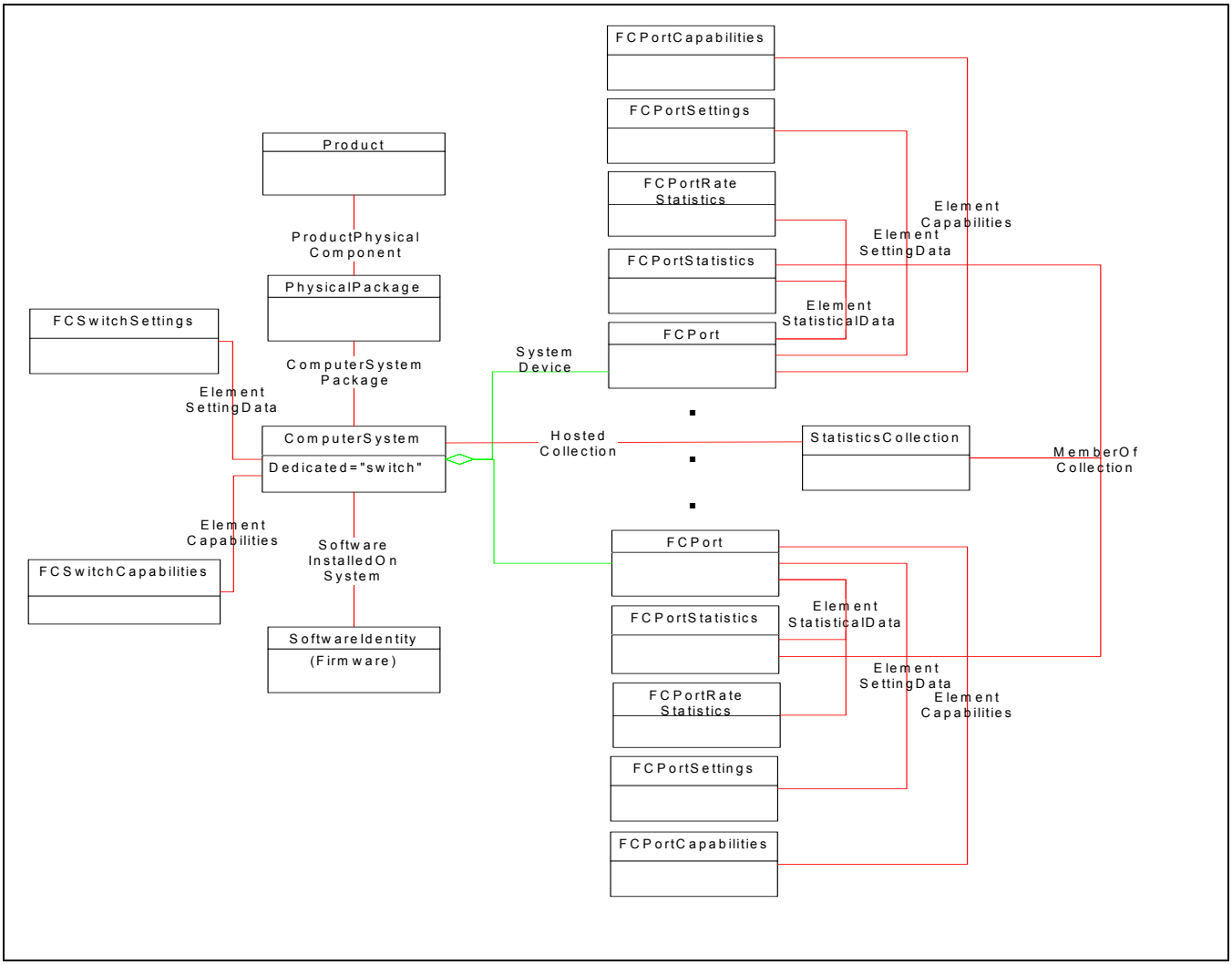

Figure 16 - Switch Instance Diagram

12.1.1 FC Port Settings and Capabilities

Capabilities describe the possible features that a ManagedElement supports. Settings are used to describe the requested configuration. The ManagedElement itself describes what settings have been applied and operating.

- For FC Port Type, there are settings that are not in the actual ManagedElement, FCPort.Types. These are settings that allow a subrange of possible port types. They are:
- A G_Port is a Switch Port that is capable of either operating as an E_Port or F_Port. A G_Port determines through Port Initialization whether it operates as an E_Port or as an F_Port.
- A GL_Port is a G_Port that is also capable of operating as an FL_Port.
- A Fx_Port is a switch port capable of operating as an F_Port or FL_Port.

The actual FCPort when operating shall run one of the port types as per FC-GS. In most cases a switch has a default setting to autonegotiate, which in most cases equates to GL or G being set in FCPortSetting.RequestedType. It is required that this setting, FCPortSetting.RequestedType, be shown regardless

of whether it was set administratively or is the default behavior of the switch. FCPortSetting.RequestedType represents a setting that the administrator can understand and clearly identify why a switch port ends up running a particular port type. If the switch does not support setting the port type, the RequestedTypesSupported array will be empty. It is valid to have a port type of “Unknown” until the link has been established. In 12.8.14 CIM_FCSwitchCapabilities, the table is showing the relationship of particular port types and how the capabilities relate to the setting for these port types.

The same concept applies for FCPort settings for speed except there is a separate property indicating auto negotiate, FCPortSettings.AutoSenseSpeed (LogicalPortSettings.AutoSenseSpeed). Note that this setting may have been previously set through some other administrative interface (e.g. CLI), but should still be reported in FCPortSettings.RequestedSpeed. If FCPortSetting.AutoSenseSpeed is true, then the value of FCPortSettings.RequestedSpeed is ignored and the speed will be negotiated by the hardware. If it is disabled, the port will operate at the speed configured in FCPortSettings.RequestedSpeed.

FCPortSettings.RequestedSpeed allows the port speed to be administratively set (WRITE qualifier). It also indicates to the client that the port has been administratively set (now or at a previous time). This property can only be set administratively if FCPortCapabilities.RequestedSpeedsSupported[] is not empty, and may only be set to one of the values in FCPortCapabilities.RequestedSpeedsSupported[].

FCPortCapabilities.RequestedSpeedsSupported indicates whether the device allows the speed to be administratively set. For instance, a 4Gb port may allow 1, 2, and 4 Gb. FCPort.Speed (LogicalPort.Speed) represents the actual speed the port is running and a speed of zero represents that the link has not been established.

12.1.2 Trunking

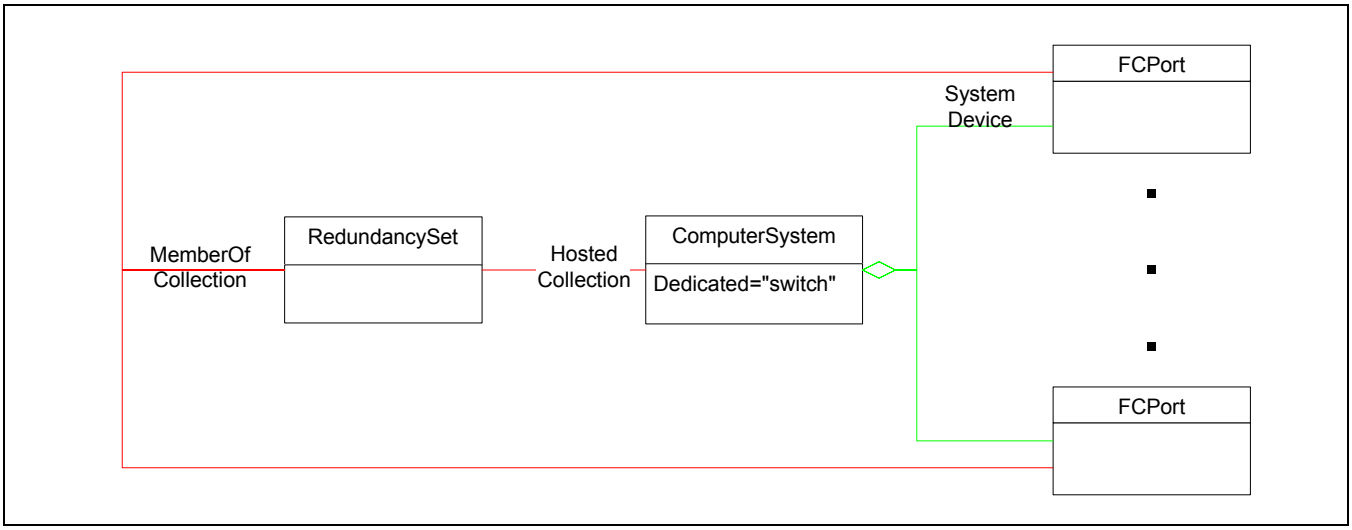

Figure 17 - Trunking Instance Diagram

Trunking describes from a switch perspective which ports are working together passing frames using the class RedundancySet. The RedundancySet has a property TypeOfSet which is used to identify what type of redundancy or trunking is occurring among the switch ports associated to the RedundancySet using MemberOfCollection.

12.2 Health and Fault Management

The following classes report possible Health and Fault information through LifeCycle indications:

- ComputerSystem
- FCPort

These LifeCycle indications are more fully described in *Storage Management Technical Specification, Part 2 Common Profiles, 1.3.0 Rev 6* Table 271.

Also in Table 110, “CIM Elements for Switch” are a list of AlertIndications which may also be indicators for Health and Fault Management.

12.3 Cascading Considerations

None

12.4 Dependencies on Profiles, Subprofiles, and Packages

Table 109 describes the supported profiles for Switch.

Table 109 - Supported Profiles for Switch

Registered Profile Names	Mandatory	Version
Blades	No	1.3.0
Access Points	No	1.3.0
Software Installation	No	1.3.0
Multiple Computer System	No	1.2.0
Switch Configuration Data	No	1.1.0
Physical Package	Yes	1.3.0
Software	Yes	1.3.0
Fabric Switch Partitioning	No	1.3.0
Power Supply	No	1.0.0
Fan	No	1.0.0
RecordLog	No	1.0.0

12.5 Methods of this Profile

None in this version of the specification

12.6 Client Considerations and Recipes

12.6.1 Enable FCPort

```
// DESCRIPTION
// This recipe describes how to enable a port on a Fibre Channel Switch.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. The instance of the port to be enabled is known as $Port.
```

```
// MAIN
// Step 1. Retrieve the capabilities of the port.
$PortCapabilities[] = Associators($Port.getObjectPath(),
     “CIM_ElementCapabilities”,
     “CIM_FCPortCapabilities”,
     “ManagedElement”,
     “Capabilities”,
     false,
     false,
     {“RequestedStatesSupported”})
if ($PortCapabilities[] == null || PortCapabilities[].length != 1) {
    <ERROR! The required port capabilities are not available>
}

// Step 2. Verify that the port can be enabled.
if (!contains(2, $PortCapabilities[0].RequestedStatesSupported)) {
    <EXIT! Enabling the specified port is not supported>
}

// Step 3. Verify that the port is in a state in which enabling is appropriate.
if ($Port.EnabledState != 2 && $Port.RequestedState == 5) {

    // Step 4. Enable the port.
    %InArguments[“RequestedState”] = 2// “Enabled”
    // Timeout request after 90 seconds
    %InArguments[“TimeoutPeriod”] = 00000000000130.000000:000
    #ReturnValue = InvokeMethod($Port.getObjectPath(),
         “RequestStateChange”,
         %InArguments,
         %OutArguments)
    if (#ReturnValue == 0) {// “Completed with No Error”
     <EXIT! Port successfully enabled>
    } else if (#ReturnValue == 4098) {// “Timeout Parameter Not Supported”
     %InArguments[“RequestedState”] = 2// “Enabled”
     %InArguments[“TimeoutPeriod”] = 0// No timeout
     #ReturnValue = InvokeMethod($Port.getObjectPath(),
        “RequestStateChange”,
        %InArguments,
        %OutArguments)
     if (#ReturnValue == 0) {// “Completed with No Error”
         <EXIT! Port successfully enabled>
     } else {
         <ERROR! Port state transition failed>
     }
    }
} else {
    <ERROR! The specified port is already enabled or currently in a
```

```
        state transition>
}
```

12.6.2 Disable Port

```
// DESCRIPTION
// This recipe describes how to disable a port on a Fibre Channel Switch.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. The instance of the port to be disabled is known as $Port.

// MAIN
// Step 1. Retrieve the capabilities of the port.
$PortCapabilities[] = Associators($Port.getObjectPath(),
     "CIM_ElementCapabilities",
     "CIM_FCPortCapabilities",
     "ManagedElement",
     "Capabilities",
     false,
     false,
     {"RequestedStatesSupported"})
if ($PortCapabilities[] == null || PortCapabilities[].length != 1) {
    <ERROR! The required port capabilities are not available>
}

// Step 2. Verify that the port can be disabled.
if (!contains(3, $Capabilities.RequestedStatesSupported)) {
    <EXIT! Disabling the specified port is not supported>
}

// Step 3. Verify that the port is in a state in which disabling is appropriate.
if ($Port.EnabledState != 3 && $Port.RequestedState == 5) {

    // Step 4. Disable the port.
    %InArguments["RequestedState"] = 3// "Disabled"
    // Timeout request after 90 seconds
    %InArguments["TimeoutPeriod"] = 00000000000130.000000:000
    #ReturnValue = InvokeMethod($Port.getObjectPath(),
         "RequestStateChange",
         %InArguments,
         %OutArguments)
    if (#ReturnValue == 0) {// "Completed with No Error"
     <EXIT! Port successfully disabled>
    } else if (#ReturnValue == 4098) {// "Timeout Parameter Not Supported"
     %InArguments["RequestedState"] = 3 // "Disabled"
     %InArguments["TimeoutPeriod"] = 0 // No timeout
     #ReturnValue = InvokeMethod($Port.getObjectPath(),
        "RequestStateChange",
```

```
            %InArguments,
            %OutArguments)
        if (#ReturnValue == 0) {// "Completed with No Error"
            <EXIT! Port successfully disabled>
        } else {
            <ERROR! Port state transition failed>
        }
       }
} else {
    <ERROR! The specified port is already disabled or currently in a
         state transition>
}
```

12.6.3 Enable Switch

```
// DESCRIPTION
// This recipe describes how to enable a Fibre Channel Switch.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. A reference to the Switch to enable is known and defined in the
//    variable $Switch->.

// MAIN
// Step 1. Retrieve the relevant Switch instance information.
$Switch = GetInstance($Switch->,
     false,
     false,
     false,
     {"EnabledState", "RequestedState"})

// Step 2. Retrieve the capabilities of the Switch.
$SwitchCapabilities[] = Associators($Switch->,
     "CIM_ElementCapabilities",
     "CIM_FCSwitchCapabilities",
     "ManagedElement",
     "Capabilities",
     false,
     false,
     {"RequestedStatesSupported"})
if ($SwitchCapabilities[] == null || SwitchCapabilities[].length != 1) {
    <ERROR! The required Switch capabilities are not available>
}

// Step 3. Verify that the Switch can be enabled.
if (!contains(2, $SwitchCapabilities[0].RequestedStatesSupported)) {
    <EXIT! Enabling the specified Switch is not supported>
}
```

```
// Step 4. Verify that the Switch is in a state in which enabling is
// appropriate.
if ($Switch.EnabledState != 2 && $Switch.RequestedState == 5) {

    // Step 5. Enable the Switch.
    %InArguments[“RequestedState”] = 2// “Enabled”
    // Timeout request after 90 seconds
    %InArguments[“TimeoutPeriod”] = 00000000000130.000000:000
    #ReturnValue = InvokeMethod($Switch->,
         “RequestStateChange”,
         %InArguments,
         %OutArguments)
    if (#ReturnValue == 0) {// “Completed with No Error”
     <EXIT! Switch successfully enabled>
    } else if (#ReturnValue == 4098) {// “Timeout Parameter Not Supported”
     %InArguments[“RequestedState”] = 2// “Enabled”
     %InArguments[“TimeoutPeriod”] = 0// No timeout
     #ReturnValue = InvokeMethod($Switch->,
        “RequestStateChange”,
        %InArguments,
        %OutArguments)
     if (#ReturnValue == 0) {// “Completed with No Error”
        <EXIT! Switch successfully enabled>
     } else {
         <ERROR! Switch state transition failed>
     }
    }
} else {
    <ERROR! The specified Switch is already enabled or currently in
         a state transition>
}
```

12.6.4 Disable Switch

```
// DESCRIPTION
// This recipe describes how to disable a Fibre Channel Switch.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. A reference to the Switch to disable is known and defined in the
//    variable $Switch->.

// MAIN
// Step 1. Retrieve the relevant Switch instance information.
$Switch = GetInstance($Switch->,
     false,
     false,
     false,
     {“EnabledState”, “RequestedState”})
```

```
// Step 2. Retrieve the capabilities of the Switch.
$SwitchCapabilities[] = Associators($Switch->,
     “CIM_ElementCapabilities”,
     “CIM_FCSwitchCapabilities”,
     “ManagedElement”,
     “Capabilities”,
     false,
     false,
     {“RequestedStatesSupported”})
if ($SwitchCapabilities[] == null || SwitchCapabilities[].length != 1) {
    <ERROR! The required Switch capabilities are not available>
}

// Step 3. Verify that the Switch can be disabled.
if (contains(3, $SwitchCapabilities[0].RequestedStatesSupported)) {
    <EXIT! Disabling the specified Switch is not supported>
}

// Step 4. Verify that the Switch is in a state in which disabling is
// appropriate.
if ($Switch.EnabledState != 3 && $Switch.RequestedState == 5) {

    // Step 5. Disable the Switch.
    %InArguments[“RequestedState”] = 3// “Disabled”
    // Timeout request after 90 seconds
    %InArguments[“TimeoutPeriod”] = 00000000000130.000000:000
    #ReturnValue = InvokeMethod($Switch->,
         “RequestStateChange”,
         %InArguments,
         %OutArguments)
    if (#ReturnValue == 0) {// “Completed with No Error”
     <EXIT! Switch successfully disabled>
    } else if (#ReturnValue == 4098) {// “Timeout Parameter Not Supported”
     %InArguments[“RequestedState”] = 3// “Disabled”
     %InArguments[“TimeoutPeriod”] = 0// No timeout
     #ReturnValue = InvokeMethod($Switch->,
        “RequestStateChange”,
        %InArguments,
        %OutArguments)
     if (#ReturnValue == 0) {// “Completed with No Error”
         <EXIT! Switch successfully disabled>
     } else {
        <ERROR! Switch state transition failed>
     }
    } else {
     <ERROR! Switch state transition failed>
```

```
    }
} else {
    <ERROR! The specified Switch is already disabled or currently in
         a state transition>
}
```

12.6.5 Reset Switch

```
// DESCRIPTION
// This recipe describes how to reset a Fibre Channel Switch.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. A reference to the Switch to reset is known and defined in the
//    variable $Switch->.

// MAIN
// Step 1. Retrieve the relevant Switch instance information.
$Switch = GetInstance($Switch->,
     false,
     false,
     false,
     {“EnabledState”, “RequestedState”})

// Step 2. Retrieve the capabilities of the Switch.
$SwitchCapabilities[] = Associators($Switch->,
     “CIM_ElementCapabilities”,
     “CIM_FCSwitchCapabilities”,
     “ManagedElement”,
     “Capabilities”,
     false,
     false,
     {“RequestedStatesSupported”})
if ($SwitchCapabilities[] == null || SwitchCapabilities[].length != 1) {
    <ERROR! The required Switch capabilities are not available>
}

// Step 3. Verify that the Switch can be reset.
if (contains(11, $SwitchCapabilities[0].RequestedStatesSupported)) {
    <EXIT! Resetting the specified Switch is not supported>
}

// Step 4. Verify that the Switch is in a state in which reseting is
// appropriate.
if ($Switch.EnabledState == 2 && $Switch.RequestedState == 5) {

    // Step 5. Reset the Switch.
    %InArguments[“RequestedState”] = 11// “Reset”
    // Timeout request after 90 seconds
```

```
    %InArguments["TimeoutPeriod"] = 00000000000130.000000:000
    #ReturnValue = InvokeMethod($Switch->,
         "RequestStateChange",
         %InArguments,
         %OutArguments)
    if (#ReturnValue == 0) {// "Completed with No Error"
     <EXIT! Switch successfully reset>
    } else if (#ReturnValue == 4098) {// "Timeout Parameter Not Supported"
     %InArguments["RequestedState"] = 11// "Reset"
     %InArguments["TimeoutPeriod"] = 0// No timeout
     #ReturnValue = InvokeMethod($Switch->,
        "RequestStateChange",
        %InArguments,
        %OutArguments)
     if (#ReturnValue == 0) {// "Completed with No Error"
         <EXIT! Switch successfully reset>
     } else {
         <ERROR! Switch state transition failed>
     }
    } else {
     <ERROR! Switch state transition failed>
    }
} else {
    <ERROR! The specified Switch is already reset or currently in
         a state transition>
}
```

12.6.6 Set Port Speed

```
// DESCRIPTION
// This recipe describes how to modify the speed of a port on a Fibre Channel
// Switch.
//
// PREEXISTING CONDITIONS AND ASSUMPTIONS
// 1. The instance of the port to whose speed to modify is known as $Port.
// 2. The desired port speed is known and defined in the variable #Speed.

// MAIN
// Step 1. Retrieve the capabilities of the port.
$PortCapabilities[] = Associators($Port.getObjectPath(),
     "CIM_ElementCapabilities",
     "CIM_FCPortCapabilities",
     "ManagedElement",
     "Capabilities",
     false,
     false,
     {"AutoSenseSpeedConfigurable", "RequestedSpeedsSupported"})
if ($PortCapabilities[] == null || PortCapabilities[].length != 1) {
```

```
    <ERROR! The required port capabilities are not available>
}
$Capabilities = $PortCapabilities[0]

// Step 2. Verify that the port speed can be set to the specified speed.
if (contains(#Speed, $Capabilities.RequestedSpeedsSupported)) {

     // Step 3. Retrieve the port settings.
     $Settings[] = Associators($Port.getObjectPath(),
        "CIM_ElementSettingData",
        "CIM_FCPortSettings",
        "ManagedSetting",
        "SettingData",
        false,
        false,
        {"InstanceID", "AutoSenseSpeed", "RequestedSpeed"})
     if ($Settings[] == null || Settings[].length != 1) {
        <ERROR! The required port settings are not available>
     }
     $PortSetting = $Settings[0]

     // Step 4. Port speed is ignored unless AutoSenseSpeed is disabled,
     if ($PortSetting.AutoSenseSpeed) {
        if ($Capabilities.AutoSenseSpeedConfigurable) {
           $PortSetting.AutoSenseSpeed = false
        } else {
           //Unlikely, but not an error
        }
     }

     // Step 5. Modify the port speed to the specified speed.
     $PortSetting.RequestedSpeed = #Speed
     ModifyInstance($PortSetting.getObjectPath(),
        $PortSetting,
        false,
        {"AutoSenseSpeed", "RequestedSpeed"})

     // Step 6. Verify that the port speed modification was applied.
     $Port = GetInstance($Port.getObjectPath(),
        false,
        false,
        false,
        {"Speed"})
     if ($Port.Speed == #Speed) {
        <EXIT! Port speed modified successfully>
     } else {
        <ERROR! Port speed was not modified as specified>
```

```
     }
} else {
     <EXIT! Specified port speed is not supported>
}
```

12.6.7 Set Port Type

```
// DESCRIPTION
// This recipe describes how to modify the port type on a Fibre Channel Switch.
//
// PREEXISTING CONDITIONS AND ASSUMPTIONS
// 1. The instance of the port to whose type to modify is known as $Port.
// 2. The desired port type is known and defined in the variable #Type.

// MAIN
// Step 1. Retrieve the capabilities of the port.
$PortCapabilities[] = Associators($Port.getObjectPath(),
     "CIM_ElementCapabilities",
     "CIM_FCPortCapabilities",
     "ManagedElement",
     "Capabilities",
     false,
     false,
     {"RequestedTypesSupported"})
if ($PortCapabilities[] == null || PortCapabilities[].length != 1) {
    <ERROR! The required port capabilities are not available>
}

// Step 2. Verify that the port type can be modified as specified.
$Capabilities = $PortCapabilities[0]
if (contains(#Type, $Capabilities.RequestedTypesSupported)) {

    // Step 3. Retrieve the port settings.
    $Settings[] = Associators($Port.getObjectPath(),
         "CIM_ElementSettingData",
         "CIM_FCPortSettings",
         "ManagedSetting",
         "SettingData",
         false,
         false,
         {"RequestedType"})
    if ($Settings[] == null || Settings[].length != 1) {
     <ERROR! The required port settings are not available>
    }
    $PortSetting = $Settings[0]

    // Step 4. Modify the port type to the specified type.
```

```
    $PortSetting.RequestedType = #Type
    ModifyInstance($PortSetting.getObjectPath(),
         $PortSetting,
         false,
         {"RequestedType"})

    // Step 5. Verify that the port type modification was applied.
    $Port = GetInstance($PortSetting.getObjectPath(),
         false,
         false,
         false,
         {"RequestedType"})
    if ($PortSetting.RequestedType == #Type) {
     <EXIT! Port type request successfully>
    }
    <ERROR! Port type request was not modified as specified>
} else {
    <ERROR! Port type request cannot be set to specified type>
}
```

12.6.8 Set Fibre Channel Switch Principal Priority

```
// DESCRIPTION
// This recipe describes how to modify the Principal Priority of a Fibre Channel
// Switch.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
//
// 1. A reference to the Switch whose Principal Priority to modify is known and
//    defined in the variable $Switch->
// 2. The desired Principal Priority of the Switch is known as #Priority.

// MAIN
// Step 1. Retrieve the capabilities of the Switch.
$SwitchCapabilities[] = Associators($Switch->,
     "CIM_ElementCapabilities",
     "CIM_FCSwitchCapabilities",
     "ManagedElement",
     "Capabilities",
     false,
     false,
     {"PrincipalPrioritiesSupported"})
if ($SwitchCapabilities[] == null || SwitchCapabilities[].length != 1) {
    <ERROR! The required Switch capabilities are not available>
}

// Step 2. Verify that the Switch Principal Priority can be modified.
$Capabilities = $SwitchCapabilities[0]
```

```
if (!contains(5, $Capabilities.PrincipalPrioritiesSupported[])) {

    $SwitchSettings[] = Associators($Switch->,
     “CIM_ElementSettingData”,
     “CIM_FCSwitchSettings”,
     “ManagedElement”,
     “SettingData”,
     false,
     false,
     {“PrincipalPriority”})
    if ($SwitchSettings[] == null || SwitchSettings[].length != 1) {
     <ERROR! Required Switch settings are not available>
    }
    $Settings = $SwitchSettings[0]

    // Step 3. Ensure a new Principal Priority is being set.
    if (#Priority != $Settings.PrincipalPriority)) {

     // Step 4. Modify the Principal Priority of the Switch.
     $Settings.PrincipalPriority = #Priority
     ModifyInstance($Settings.getObjectPath(),
        $Settings,
        false,
        {“PrincipalPriority”})
     // Step 5. Verify that the Switch priority modification was applied.
     $Settings = GetInstance($Settings.getObjectPath(),
        false,
        false,
        false,
        {“PrincipalPriority”})
     if ($Settings.PrincipalPriority == #Priority) {
         <EXIT! Switch Principal Priority was modified successfully>
     }
     <EXIT! Switch Principal Priority was not modified successfully>
    } else {
     <ERROR! Principal Priority specified is already set>
    }
} else {
    // “Not Applicable”
    <EXIT! The Switch does not support Principal Priority modification>
}
```

12.6.9 Set Switch Name

```
// DESCRIPTION
// This recipe describes how to modify the name of a Fibre Channel Switch.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
```

```
//
// 1. A reference to the Switch whose name to modify is known and defined in
//    the variable $Switch->
// 2. The desired name of the Switch is known as #Name.

// MAIN
// Step 1. Retrieve the capabilities of the Switch.
$SwitchCapabilities[] = Associators($Switch->,
     "CIM_ElementCapabilities",
     "CIM_FCSwitchCapabilities",
     "ManagedElement",
     "Capabilities",
     false,
     false,
     {"ElementNameEditSupported", "MaxElementNameLen"})
if ($SwitchCapabilities[] == null || $SwitchCapabilities[].length != 1) {
    <ERROR! The required Switch capabilities are not available>
}

// Step 2. Verify that the Switch name can be modified.
$Capabilities = $SwitchCapabilities[0]
if ($Capabilities.ElementNameEditSupported) {

    // Step 3. Verify that the new name to be specified is within the
    // constraints of the name length supported by the Switch.
    if (#Name.length() < $Capabilities.MaxElementNameLen) {

     // Step 4. Retrieve the instance representing the Switch.
     $Switch = GetInstance($Switch->,
        false,
        false,
        false,
        {"ElementName"})

     // Step 5. Modify the name of the Switch.
     $Switch.ElementName = #Name
     ModifyInstance($Switch->,
        $Switch,
        false,
        {"ElementName"})

     // Step 6. Verify that the Switch name change was applied.
     $Switch = GetInstance($Switch->,
        false,
        false,
        false,
        {"ElementName"})
```

```
     if (compare(#Name, $Switch.ElementName)) {
         <EXIT! Switch name was modified successfully>
     }
     <ERROR! Switch name was not modified successfully>
    }
    <ERROR! Specified Switch name exceeds length limit>
} else {
    <EXIT! The Switch does not support name modification>
}
```

12.6.10 Set Port Name

```
// DESCRIPTION
// This recipe describes how to modify the name of a Port on a Fibre Channel
// Switch.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. The instance of the port to whose type to modify is known as $Port.
// 2. The desired name of the port is known as #Name.

// MAIN
// Step 1. Retrieve the capabilities of the port.
$PortCapabilities[] = Associators($Port.getObjectPath(),
     "CIM_ElementCapabilities",
     "CIM_FCPortCapabilities",
     "ManagedElement",
     "Capabilities",
     false,
     false,
     {"ElementNameEditSupported", "MaxElementNameLen"})
if ($PortCapabilities[] == null || $PortCapabilities[].length != 1) {
    <ERROR! The required Port capabilities are not available>
}

// Step 2. Verify that the port name can be modified.
$Capabilities = $PortCapabilities[0]
if ($Capabilities.ElementNameEditSupported) {

    // Step 3. Verify that the new name to be specified is within the
    // constraints of the name length supported by the port.
    if (#Name.length() < $Capabilities.MaxElementNameLen) {

     // Step 4. Modify the name of the port.
     $Port.ElementName = #Name
     ModifyInstance($Port.getObjectPath(),
        $Port,
        false,
        {"ElementName"})
```

```
     // Step 5. Verify that the port name change was applied.
     $Port = GetInstance($Port.getObjectPath(),
        false,
        false,
        false,
        {"ElementName"})
     if (compare(#Name, $Port.ElementName)) {
         <EXIT! Port name was modified successfully>
     }
     <ERROR! Port name was not modified successfully>
    }
    <ERROR! Specified Port name exceeds length limit>
} else {
    <EXIT! The Port does not support name modification>
}
```

12.6.11 Set Fibre Channel Switch Preferred Domain ID

```
// DESCRIPTION
//
// This recipe describes how to modify the preferred Domain ID of a Switch.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. A reference to the Switch to reset is known and defined in the
//    variable $Switch->.
// 2. The new preferred Domain ID to be set on the Switch is known as #DomainID.

// MAIN
// Step 1. Retrieve the capabilities of the Switch.
$SwitchCapabilities[] = Associators($Switch->,
     "CIM_ElementCapabilities",
     "CIM_FCSwitchCapabilities",
     "ManagedElement",
     "Capabilities",
     false,
     false,
     {"DomainIDConfigureable", "MinDomainID", "MaxDomainID"})
if ($SwitchCapabilities[] == null || SwitchCapabilities[].length != 1) {
    <ERROR! The required Switch capabilities are not available>
}

// Step 2. Verify that the Switch's preferred Domain ID can be modified.
$Capabilities = $SwitchCapabilities[0]
if ($Capabilities.DomainIDConfigureable) {

    // Step 3. Verify that the desired Domain ID is within the permissible
    // range.
```

```
if (#DomainID >= $Capabilities.MinDomainID
     && #DomainID <= $Capabilities.MaxDomainID) {

 // Step 4. Retrieve the Switch settings.
 $Settings[] = Associators($Switch->,
    "CIM_ElementSettingData",
    "CIM_FCSwitchSettings",
    "ManagedSetting",
    "SettingData",
    false,
    false,
    {"PreferredDomainID"})
 if ($Settings[] == null || Settings[].length != 1) {
     <ERROR! The required Switch settings are not available>
 }
 $SwitchSetting = $Settings[0]

 // Step 5. Modify the Switch Domain ID to the specified preferred value.
 $SwitchSetting.PreferredDomainID = #DomainID
 ModifyInstance($SwitchSetting.getObjectPath(),
    $SwitchSetting,
    false,
    {"PreferredDomainID"})

 // Step 6. Verify that the Switch Domain ID modification was applied.
 $Switch = GetInstance($Switch->,
    false,
    false,
    false,
    {"IdentifyingDescriptions", "OtherIdentifyingInfo"})
 // NOTE: The Domain ID value is contained in the OtherIdentifyingInfo
 // property at the same index as the "DomainID" element index in the
 // IdentifyingDescriptions property.
 #index = -1
 while (#i < $Switch.IdentifyingDescriptions[].length
    && #index < 0) {
     if ($Switch.IdentifyingDescriptions[#i] == "DomainID") {
    #index = #i
     }
 }
 if (#index >= 0 && $Switch.OtherIdentifyingInfo[#index] == #DomainID) {
     <EXIT! Switch Domain ID successfully modified>
 }
 <ERROR! Switch Domain ID was not modified as specified>
} else {
 <ERROR! Domain ID specified is not within permitted range>
}
```

```
} else {
    <EXIT! Domain ID configuration on the specified Switch is not supported>
}
```

12.6.12 Lock Fibre Channel Switch Domain ID

```
// DESCRIPTION
//
// This recipe describes how to set the Domain ID Lock of a Switch.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. A reference to the Switch whose Domain ID to lock is known and defined
// in the variable $Switch->.

// MAIN
// Step 1. Retrieve the capabilities of the Switch.
$SwitchCapabilities[] = Associators($Switch->,
     “CIM_ElementCapabilities”,
     “CIM_FCSwitchCapabilities”,
     “ManagedElement”,
     “Capabilities”,
     false,
     false,
     {“DomainIDLockedSupported”})
if ($SwitchCapabilities[] == null || $SwitchCapabilities[].length != 1) {
    <ERROR! The required Switch capabilities are not available>
}

// Step 2. Verify that the Switch’s Domain ID Lock can be set.
$Capabilities = $SwitchCapabilities[0]
if ($Capabilities.DomainIDLockedSupported) {

    // Step 3. Retrieve the Switch settings.
    $Settings[] = Associators($Switch->,
         “CIM_ElementSettingData”,
         “CIM_FCSwitchSettings”,
         “ManagedSetting”,
         “SettingData”,
         false,
         false,
         {“DomainIDLocked”, “PreferredDomainID”})
    if ($Settings[] == null || Settings[].length != 1) {
     <ERROR! The required Switch settings are not available>
    }
    $SwitchSetting = $Settings[0]
    #PreferredDomainID = $SwitchSetting.PreferredDomainID

    // Step 4. Verify that the Domain ID is not already locked.
```

```
        if ($SwitchSetting.DomainIDLocked) {
         <EXIT! The Domain ID Lock is already set>
        }

        // Step 5. Lock the Switch Domain ID.
        $SwitchSetting.DomainIDLocked = true
        ModifyInstance($SwitchSetting.getObjectPath(),
             $SwitchSetting,
             false,
             {"DomainIDLocked"})

        // Step 6. Verify that the Switch Domain ID specifies the preferred
        // Domain ID.
        $Switch = GetInstance($Switch->,
             false,
             false,
             false,
             {"IdentifyingDescriptions", "OtherIdentifyingInfo"})
        // NOTE: The Domain ID value is contained in the OtherIdentifyingInfo
        // property at the same index as the "DomainID" element index in the
        // IdentifyingDescriptions property.
        #index = -1
        while (#i < $Switch.IdentifyingDescriptions[].length && #index < 0) {
         if ($Switch.IdentifyingDescriptions[#i] == "DomainID") {
             #index = #i
         }
        }
        if (#index >= 0 &&
             $Switch.OtherIdentifyingInfo[#index] == #PreferredDomainID) {
         <EXIT! Switch Domain ID successfully locked>
        }
        <ERROR! Switch Domain ID does not reflect the preferred Domain ID>
    } else {
        <EXIT! Domain ID configuration on the specified Switch is not supported>
    }
```

12.7 Registered Name and Version

Switch version 1.3.0

12.8 CIM Elements

Table 109 describes the CIM elements for Switch.

Table 110 - CIM Elements for Switch

Element Name	Requirement	Description
12.8.1 CIM_ComputerSystem (Switch)	Mandatory	Represents the Switch
12.8.2 CIM_ComputerSystemPackage	Mandatory	Associates PhysicalPackage to the ComputerSystem (Switch)
12.8.3 CIM_ElementCapabilities (FCPort to FCPortCapabilities)	Mandatory	Associates FCPort to the FcPortCapabilities
12.8.4 CIM_ElementCapabilities (System to FCSwitchCapabilities)	Mandatory	Associates FCSwitchCapabilities to the ComputerSystem (Switch)
12.8.5 CIM_ElementSettingData (FCPortSettings to FCPort)	Optional	Associates FCPortSettings to FcPort
12.8.6 CIM_ElementSettingData (FCSwitchSettings to ComputerSystem)	Conditional	Conditional requirement: Support for the Switch Configuration Data profile.Associates FCSwitchSettings to ComputerSystem (Switch)
12.8.7 CIM_ElementStatisticalData (FCPortRateStatistics to FCPort)	Optional	Associates FCPortRateStatistics to the FCPort
12.8.8 CIM_ElementStatisticalData (FCPortStatistics to FCPort)	Mandatory	Associates FCPortStatistics to the FCPort
12.8.9 CIM_FCPort	Mandatory	Fibre Channel Switch Port
12.8.10 CIM_FCPortCapabilities	Mandatory	Switch Port Capabilities
12.8.11 CIM_FCPortRateStatistics	Optional	Fibre Channel Switch Port Rate Statistics
12.8.12 CIM_FCPortSettings	Optional	Switch Port Settings
12.8.13 CIM_FCPortStatistics	Mandatory	Fibre Channel Switch Port Statistics.
12.8.14 CIM_FCSwitchCapabilities	Mandatory	Fibre Channel Switch Capabilities
12.8.15 CIM_FCSwitchSettings	Mandatory	Fibre Channel Switch Settings
12.8.16 CIM_HostedCollection	Mandatory	Associates the Statistics Collection to the ComputerSystem representing the switch.
12.8.17 CIM_MemberOfCollection (FcPort to REdundancySet)	Optional	Associates the FCPort to the RedundancySet.
12.8.18 CIM_MemberOfCollection (NetworkPortStatistics to StatisticalCollection)	Mandatory	Associates the NetworkPortStatistics to the StatisticsCollection.
12.8.19 CIM_ProtocolEndpoint	Optional	The endpoint of a link (ActiveConnection).
12.8.20 CIM_RedundancySet	Optional	The class RedundancySet along with the association MemberOfCollection in this profile is used to show port aggregation for Fibre Channel trunking.

Table 110 - CIM Elements for Switch

Element Name	Requirement	Description
12.8.21 CIM_StatisticsCollection	Mandatory	Collection to aggregate the FCPortStatistics for each switch
12.8.22 CIM_SystemDevice	Mandatory	Associates FCPort to the ComputerSystem (Switch)
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_ComputerSystem	Mandatory	New Switch Instance
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_ComputerSystem	Mandatory	Deletion of Switch Instance
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND SourceInstance.OperationalStatus <> PreviousInstance.OperationalStatus	Mandatory	Deprecated WQL -Modification of OperationalStatus in Switch Instance
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.OperationalStatus <> PreviousInstance.OperationalStatus	Mandatory	Deprecated WQL -Modification of OperationalStatus in FC Port Instance
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND SourceInstance.CIM_ComputerSystem::OperationalStatus <> PreviousInstance.CIM_ComputerSystem::OperationalStatus	Mandatory	CQL -Modification of OperationalStatus in Switch Instance
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_FCPort AND SourceInstance.CIM_FCPort::OperationalStatus <> PreviousInstance.CIM_FCPort::OperationalStatus	Mandatory	CQL -Modification of OperationalStatus in FC Port Instance

12.8.1 CIM_ComputerSystem (Switch)

Represents the Switch

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 111 describes class CIM_ComputerSystem (Switch).

Table 111 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Switch)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	Name of Class
Name	D	Mandatory	Switch Name (WWN)
ElementName		Optional	User friendly name. Can be set if FCSwitchCapabilities.ElementNameEditSupported for the switch is True.
NameFormat		Mandatory	Shall be 'WWN'.
OtherIdentifyingInfo	C	Optional	DomainID stored in decimal format, with a value between 0 and 255.
OperationalStatus		Mandatory	
IdentifyingDescriptions		Optional	'DomainID' is placed into corresponding index of OtherIdentifyingInfo
Dedicated		Mandatory	'Switch'
EnabledState		Optional	See Table ...
RequestedState		Mandatory	The Switch state requested via RequestStateChange(). Shall be of the range specified in FCSwitchCapabilities.RequestedStatesSupported if a state change has been requested. Otherwise shall be 'Not Applicable'.
EnabledDefault		Optional	Default startup for the Switch
RequestStateChange()		Optional	

12.8.2 CIM_ComputerSystemPackage

Associates PhysicalPackage to the ComputerSystem (Switch)

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

Table 112 describes class CIM_ComputerSystemPackage.

Table 112 - SMI Referenced Properties/Methods for CIM_ComputerSystemPackage

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	The reference to the PhysicalPackage
Dependent		Mandatory	The reference to the ComputerSystem

12.8.3 CIM_ElementCapabilities (FCPort to FCPortCapabilities)

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

Table 113 describes class CIM_ElementCapabilities (FCPort to FCPortCapabilities).

Table 113 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (FCPort to FCPort-Capabilities)

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	The reference to the FcPort
Capabilities		Mandatory	The reference to the FCPortCapabilities

12.8.4 CIM_ElementCapabilities (System to FCSwitchCapabilities)

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

Table 114 describes class CIM_ElementCapabilities (System to FCSwitchCapabilities).

Table 114 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (System to FCSwitchCapabilities)

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	The reference to the ComputerSystem
Capabilities		Mandatory	The reference to the FCSwitchCapabilities

12.8.5 CIM_ElementSettingData (FCPortSettings to FCPort)

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

Table 115 describes class CIM_ElementSettingData (FCPortSettings to FCPort).

Table 115 - SMI Referenced Properties/Methods for CIM_ElementSettingData (FCPortSettings to FCPort)

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	The reference to the FCPort
SettingData		Mandatory	The reference to the FCPortSettings

12.8.6 CIM_ElementSettingData (FCSwitchSettings to ComputerSystem)

Associates FCSwitchSettings to ComputerSystem (Switch)

Created By: Static
Modified By: Static
Deleted By: Static
Requirement: Support for the Switch Configuration Data profile.

Table 116 describes class CIM_ElementSettingData (FCSwitchSettings to ComputerSystem).

Table 116 - SMI Referenced Properties/Methods for CIM_ElementSettingData (FCSwitchSettings to ComputerSystem)

Properties	Flags	Requirement	Description & Notes
SettingData		Mandatory	The reference to the FCSwitchSettings
ManagedElement		Mandatory	The reference to the ComputerSystem

12.8.7 CIM_ElementStatisticalData (FCPortRateStatistics to FCPort)

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

Table 117 describes class CIM_ElementStatisticalData (FCPortRateStatistics to FCPort).

Table 117 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (FCPortRateStatistics to FCPort)

Properties	Flags	Requirement	Description & Notes
Stats		Mandatory	The reference to the FCPortRateStatistics
ManagedElement		Mandatory	The reference to the FCPort

12.8.8 CIM_ElementStatisticalData (FCPortStatistics to FCPort)

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

Table 118 describes class CIM_ElementStatisticalData (FCPortStatistics to FCPort).

Table 118 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (FCPortStatistics to FCPort)

Properties	Flags	Requirement	Description & Notes
Stats		Mandatory	The reference to the FCPortStatistics
ManagedElement		Mandatory	The reference to the FCPort

12.8.9 CIM_FCPort

The Fibre Channel Switch Port.

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

Table 119 describes class CIM_FCPort.

Table 119 - SMI Referenced Properties/Methods for CIM_FCPort

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	The scoping System's CreationClassName.
SystemName		Mandatory	The scoping System's Name.
CreationClassName		Mandatory	The Class Name
DeviceID		Mandatory	Opaque
ElementName		Mandatory	User friendly name. Can be set if FCPortCapabilities.ElementNameEditSupported is True.
OperationalStatus		Mandatory	See ...
EnabledState		Optional	See ...
RequestedState		Mandatory	The port state requested via RequestStateChange(). Shall be of the range specified in FCPortCapabilities.RequestedStatesSupported if a state change has been requested. Otherwise shall be 'Not Applicable'.
EnabledDefault		Mandatory	Default startup for the port. Used in conjunction with RequestedState can allow for persistent disabling of a port.

Table 119 - SMI Referenced Properties/Methods for CIM_FCPort

Properties	Flags	Requirement	Description & Notes
Speed		Mandatory	Speed of zero represents a link not established. 1Gb is 1062500000 bps 2Gb is 2125000000 bps 4Gb is 4250000000 bps 10Gb single channel variants are 10518750000 bps 10Gb four channel variants are 12750000000 bps This is the raw bit rate.
MaxSpeed		Mandatory	The max speed of the Port in Bits per Second using the same algorithm as Speed.
PortType		Mandatory	FC-GS Port.Type The specific mode currently enabled for the Port. The values: 'N' = Node Port 'NL' = Node Port supporting FC arbitrated loop 'E' = Expansion Port connecting fabric elements (for example, FC switches) 'F' = Fabric (element) Port 'FL' = Fabric (element) Port supporting FC arbitrated loop 'B' = Bridge Port. PortTypes are defined in the ANSI INCITS FC-GS standards. Can be set using FCPortSettings.RequestedType.
PortNumber		Mandatory	NetworkPorts are often numbered relative to either a logical modules or a network element.
PermanentAddress		Mandatory	Fibre Channel Port WWN.
LinkTechnology		Mandatory	'FC'
RequestStateChange ()		Optional	Method to change the port state. FCPortCapabilities.RequestedStatesSupported indicates what states can be set.

12.8.10 CIM_FCPortCapabilities

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

Table 120 describes class CIM_FCPortCapabilities.

Table 120 - SMI Referenced Properties/Methods for CIM_FCPortCapabilities

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	Shall be 'FC Port Capabilities'
ElementNameEditSupported		Mandatory	Indicates whether FCPort.ElementName is settable
MaxElementNameLen		Mandatory	Indicates the maximum string length of FCPort.ElementName
RequestedStatesSupported		Mandatory	Indicates the supported states for calling FCPort.RequestStateChange().
RequestedSpeedsSupported		Mandatory	Indicates the supported speeds that can be set in FCPortSettings.RequestedSpeed
AutoSenseSpeedConfigurable		Mandatory	Indicates whether FCPortSettings.AutoSenseSpeed can be set to auto-negotiate speed.
RequestedTypesSupported		Mandatory	Indicates the list of supported port types that can be set in FCPortSettings.RequestedType.

12.8.11 CIM_FCPortRateStatistics

Fibre Channel Switch Port Rate Statistics represent the rate per second over the SampleInterval. An instance of this class can represent the statistics for the current statistics, archived and consolidated statistics, or both.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Optional

Table 121 describes class CIM_FCPortRateStatistics.

Table 121 - SMI Referenced Properties/Methods for CIM_FCPortRateStatistics

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
StatisticTime		Mandatory	The time the statistic was collected.
SampleInterval		Mandatory	The interval at which the rates are calculated
TxFrameRate		Optional	
RxFrameRate		Optional	
MaxTxFrameRate		Optional	
MaxRxFrameRate		Optional	
TxRate		Mandatory	

Table 121 - SMI Referenced Properties/Methods for CIM_FCPortRateStatistics

Properties	Flags	Requirement	Description & Notes
RxRate		Mandatory	
PeakTxRate		Optional	
PeakRxRate		Optional	

12.8.12 CIM_FCPortSettings

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

Table 122 describes class CIM_FCPortSettings.

Table 122 - SMI Referenced Properties/Methods for CIM_FCPortSettings

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	Shall be 'FC Port Settings'
RequestedSpeed	M	Mandatory	The requested value to which FCPort.Speed should be set
AutoSenseSpeed	M	Mandatory	The request for the FCPort to auto sense the speed (FCPort.Speed).
RequestedType	M	Mandatory	The requested setting for the FCPort.PortType.

12.8.13 CIM_FCPortStatistics

Snapshot of performance and error counters for the Fibre Channel Switch.

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

Table 123 describes class CIM_FCPortStatistics.

Table 123 - SMI Referenced Properties/Methods for CIM_FCPortStatistics

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
StatisticTime		Optional	The time the statistics were collected. If historical data is instantiated (present), this property shall be set with the time representing the time the statistic was collected.

Table 123 - SMI Referenced Properties/Methods for CIM_FCPortStatistics

Properties	Flags	Requirement	Description & Notes
ElementName		Optional	
BytesTransmitted		Mandatory	
BytesReceived		Mandatory	
PacketsTransmitted		Mandatory	
PacketsReceived		Mandatory	
CRCErrors		Mandatory	
LinkFailures		Mandatory	
PrimitiveSeqProtocol ErrCount		Mandatory	
LIPCount		Optional	
NOSCount		Optional	
ErrorFrames		Optional	
DumpedFrames		Optional	
LossOfSignalCounter		Optional	
LossOfSyncCounter		Optional	
InvalidTransmission Words		Optional	
FramesTooShort		Optional	
FramesTooLong		Optional	
AddressErrors		Optional	
BufferCreditNotProvi ded		Optional	
BufferCreditNotRecei ved		Optional	
DelimiterErrors		Optional	
EncodingDisparityErr ors		Optional	
LinkResetsReceived		Optional	
LinkResetsTransmitt ed		Optional	
MulticastFramesRec eived		Optional	
MulticastFramesTran smitted		Optional	

Table 123 - SMI Referenced Properties/Methods for CIM_FCPortStatistics

Properties	Flags	Requirement	Description & Notes
FBSYFrames		Optional	
PBSYFrames		Optional	
FRJTFrames		Optional	
PRJTFrames		Optional	
RXClass1Frames		Optional	
TXClass1Frames		Optional	
Class1FBSY		Optional	
Class1PBSY		Optional	
Class1FRJT		Optional	
Class1PRJT		Optional	
RXClass2Frames		Optional	
TXClass2Frames		Optional	
Class2FBSY		Optional	
Class2PBSY		Optional	
Class2FRJT		Optional	
Class2PRJT		Optional	
RXClass3Frames		Optional	
TXClass3Frames		Optional	
Class3FramesDiscarded		Optional	
RXBroadcastFrames		Optional	
TXBroadcastFrames		Optional	

12.8.14 CIM_FCSwitchCapabilities

Fibre Channel Switch Capabilities

EXPERIMENTAL |

When the attached HBA supports FC-SB, and the switch is in a cascaded configuration, DomainIDConfigureable and DomainIDLockedSupported shall be TRUE.

| EXPERIMENTAL

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 124 describes class CIM_FCSwitchCapabilities.

Table 124 - SMI Referenced Properties/Methods for CIM_FCSwitchCapabilities

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	Shall be 'FC Switch Capabilities'.
ElementNameEditSupported		Mandatory	Capability indicating whether ComputerSystem.ElementName for the switch can be set.
MaxElementNameLen		Mandatory	Capability specifying the maximum name of ComputerSystem.ElementName for the switch
RequestedStatesSupported		Mandatory	The states the switch can support via ComputerSystem.RequestedState.
DomainIDConfigureable		Mandatory	Indicates whether the DomainID setting can be modified.
MinDomainID		Conditional	Conditional requirement: Support for DomainID configuration.Shall be set if DomainIDConfigureable is true.
MaxDomainID		Conditional	Conditional requirement: Support for DomainID configuration.Shall be set if DomainIDConfigureable is true.
DomainIDLockedSupported		Mandatory	
PrincipalPrioritiesSupported		Mandatory	

12.8.15 CIM_FCSwitchSettings

Fibre Channel Switch Settings

EXPERIMENTAL |

When the attached HBA supports FC-SB, and the switch is in a cascaded configuration, PreferredDomainID shall be set to a unique value, and DomainIDLocked shall be true.

| EXPERIMENTAL

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 125 describes class CIM_FCSwitchSettings.

Table 125 - SMI Referenced Properties/Methods for CIM_FCSwitchSettings

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	Shall be 'FC Switch Settings'.
PreferredDomainID	M	Conditional	Conditional requirement: Support for DomainID configuration.Required if FCSwitchCapabilities.DomainIDConfigureable is TRUE.
DomainIDLocked	M	Conditional	Conditional requirement: Support for DomainID locking.Required if FCSwitchCapabilities.DomainIDLockSupported is TRUE.
PrincipalPriority	M	Conditional	Conditional requirement: Support for Principal Priorities.Required if FCSwitchCapabilities.PrincipalPrioritiesSupported is not set to 'Not Applicable'.

12.8.16 CIM_HostedCollection

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

Table 126 describes class CIM_HostedCollection.

Table 126 - SMI Referenced Properties/Methods for CIM_HostedCollection

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

12.8.17 CIM_MemberOfCollection (FcPort to REdundancySet)

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

Table 127 describes class CIM_MemberOfCollection (FcPort to REdundancySet).

Table 127 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (FcPort to REdundancySet)

Properties	Flags	Requirement	Description & Notes
Member		Mandatory	The reference to the FcPort
Collection		Mandatory	The reference to the RedundancySet

12.8.18 CIM_MemberOfCollection (NetworkPortStatistics to StatisticalCollection)

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

Table 128 describes class CIM_MemberOfCollection (NetworkPortStatistics to StatisticalCollection).

Table 128 - SMI Referenced Properties/Methods for CIM_MemberOfCollection (NetworkPortStatistics to StatisticalCollection)

Properties	Flags	Requirement	Description & Notes
Member		Mandatory	The reference to the FcPortStatistics
Collection		Mandatory	The reference to the StatisticsColelction

12.8.19 CIM_ProtocolEndpoint

The endpoint of a link (ActiveConnection). ProtocolEndpoint shall be implemented when BroadcastReset() is supported (Force LIP). It is expected that the Fabric Profile is also implemented which defines the necessary information for determining who will receive the Force LIP on the loop.

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

Table 129 describes class CIM_ProtocolEndpoint.

Table 129 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	The scoping System's CreationClassName.
SystemName		Mandatory	The scoping System's Name.
CreationClassName		Mandatory	Name of Class
Name	CD	Mandatory	The Fibre Channel Port WWN.

Table 129 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint

Properties	Flags	Requirement	Description & Notes
NameFormat		Mandatory	'WWN'
ProtocolIFType		Mandatory	Shall be 56 (Fibre Channel).
BroadcastResetSupported		Mandatory	
BroadcastReset()		Optional	Sends a Force LIP to all attached Ports. Required if BroadcastResetSupported is TRUE.

12.8.20 CIM_RedundancySet

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

Table 130 describes class CIM_RedundancySet.

Table 130 - SMI Referenced Properties/Methods for CIM_RedundancySet

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
TypeOfSet		Mandatory	

12.8.21 CIM_StatisticsCollection

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

Table 131 describes class CIM_StatisticsCollection.

Table 131 - SMI Referenced Properties/Methods for CIM_StatisticsCollection

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	
SampleInterval		Mandatory	
TimeLastSampled		Mandatory	

12.8.22 CIM_SystemDevice

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

Table 132 describes class CIM_SystemDevice.

Table 132 - SMI Referenced Properties/Methods for CIM_SystemDevice

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	The reference to the FcPort
GroupComponent		Mandatory	The reference to the System

STABLE

EXPERIMENTAL

Clause 13: Switch Configuration Data Subprofile

13.1 Description

This subprofile describes the ability to retrieve a configuration from a switch and latter apply that configuration back on the switch (similar to an image backup and restoration of a computer system).

The profile only has three classes providing all the functionality. When a client needs to obtain a snapshot of the switch configuration, he enumerates ConfigurationData which will return the current configuration with the timestamp set appropriately.

When the client wants to apply a configuration, the client creates an instance of ConfigurationData and calls the method ApplyConfiguration() on the instance containing the property ConfigurationInformation which is to be applied to the switch.

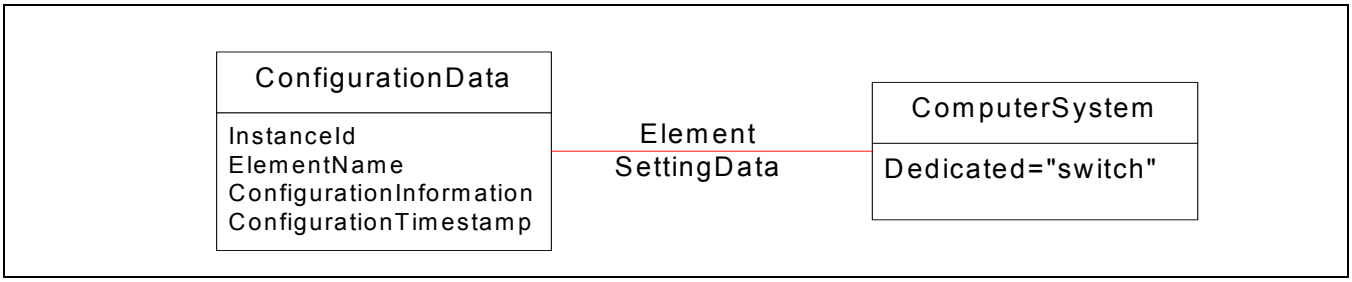

Figure 18 - Switch Configuration Data Instance

13.2 Durable Names and Correlatable IDs of the Profile

None in this version of the specification.

13.3 Instrumentation Requirements

None in this version of the specification.

13.4 Health and Fault Management

None

13.5 Cascading Considerations

None

13.6 Methods of this Profile

13.6.1 ApplyConfiguration

This method applies the configuration data to the switch. The data in the instance's ConfigurationInformation property is used as the configuration to apply. Note that it is not necessary for the element to be associated with the ConfigurationData instance at the time that this method is called.

```
uint32 ApplyConfiguration (
```

```
    boolean ValidateOnly,
    uint16 TypeOfConfiguration
    CIM_ManagedElement REF ManagedElement);
```

13.7 Client Considerations and Recipes

13.7.1 Get Switch Configuration

```
// DESCRIPTION
//
// This recipe describes how to retrieve Switch configuration data.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTIONS
// 1. A reference to the Switch whose configuration data to retrieve is known
// and defined in the variable $Switch->.

// MAIN
// Step 1. Retrieve the configuration of the Switch.
$ConfigData[] = Associators($Switch->,
     "CIM_ElementSettingData",
     "CIM_ConfigurationData",
     "ManagedElement",
     "SettingData",
     false,
     false,
     {"ConfigurationInformation", "ConfigurationTimestamp"})
if ($ConfigData[] == null || $ConfigData[].length != 1) {
    <ERROR! The required Switch configuration data is not available>
}
$SwitchConfig = $ConfigData[0]
```

13.7.2 Set Switch Configuration

```
// DESCRIPTION
//
// Set Switch Configuration
//
// PREEXISTING CONDITIONS AND ASSUMPTIONS
//
// None

Placeholder File
```

13.8 Registered Name and Version

Switch Configuration Data version 1.1.0

13.9 CIM Elements

Table 133 describes the CIM elements for Switch Configuration Data.

Table 133 - CIM Elements for Switch Configuration Data

Element Name	Requirement	Description
13.9.1 CIM_ComputerSystem	Mandatory	Represents the Switch
13.9.2 CIM_ConfigurationData	Mandatory	Switch Configuration Data
13.9.3 CIM_ElementSettingData	Mandatory	Associates ConfigurationData to the switch

13.9.1 CIM_ComputerSystem

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

Table 134 describes class CIM_ComputerSystem.

Table 134 - SMI Referenced Properties/Methods for CIM_ComputerSystem

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	The class name
Name		Mandatory	Switch Name (WWN)
NameFormat		Mandatory	"WWN"
Dedicated		Mandatory	"Switch"

13.9.2 CIM_ConfigurationData

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

Table 135 describes class CIM_ConfigurationData.

Table 135 - SMI Referenced Properties/Methods for CIM_ConfigurationData

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	User friendly for configuration file.

Table 135 - SMI Referenced Properties/Methods for CIM_ConfigurationData

Properties	Flags	Requirement	Description & Notes
ConfigurationInformation		Mandatory	The configuration data of the switch.
ConfigurationTimestamp		Mandatory	Time the configuration data was obtained
ApplyConfiguration()		Mandatory	Method that processes the configuration in the same instance and applies it to the switch

13.9.3 CIM_ElementSettingData

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

Table 136 describes class CIM_ElementSettingData.

Table 136 - SMI Referenced Properties/Methods for CIM_ElementSettingData

Properties	Flags	Requirement	Description & Notes
SettingData		Mandatory	
ManagedElement		Mandatory	

EXPERIMENTAL

STABLE

Clause 14: Blades Subprofile

14.1 Description

This subprofile describes how blades in a director class switch can be discovered and managed.

14.1.1 Instance Diagram

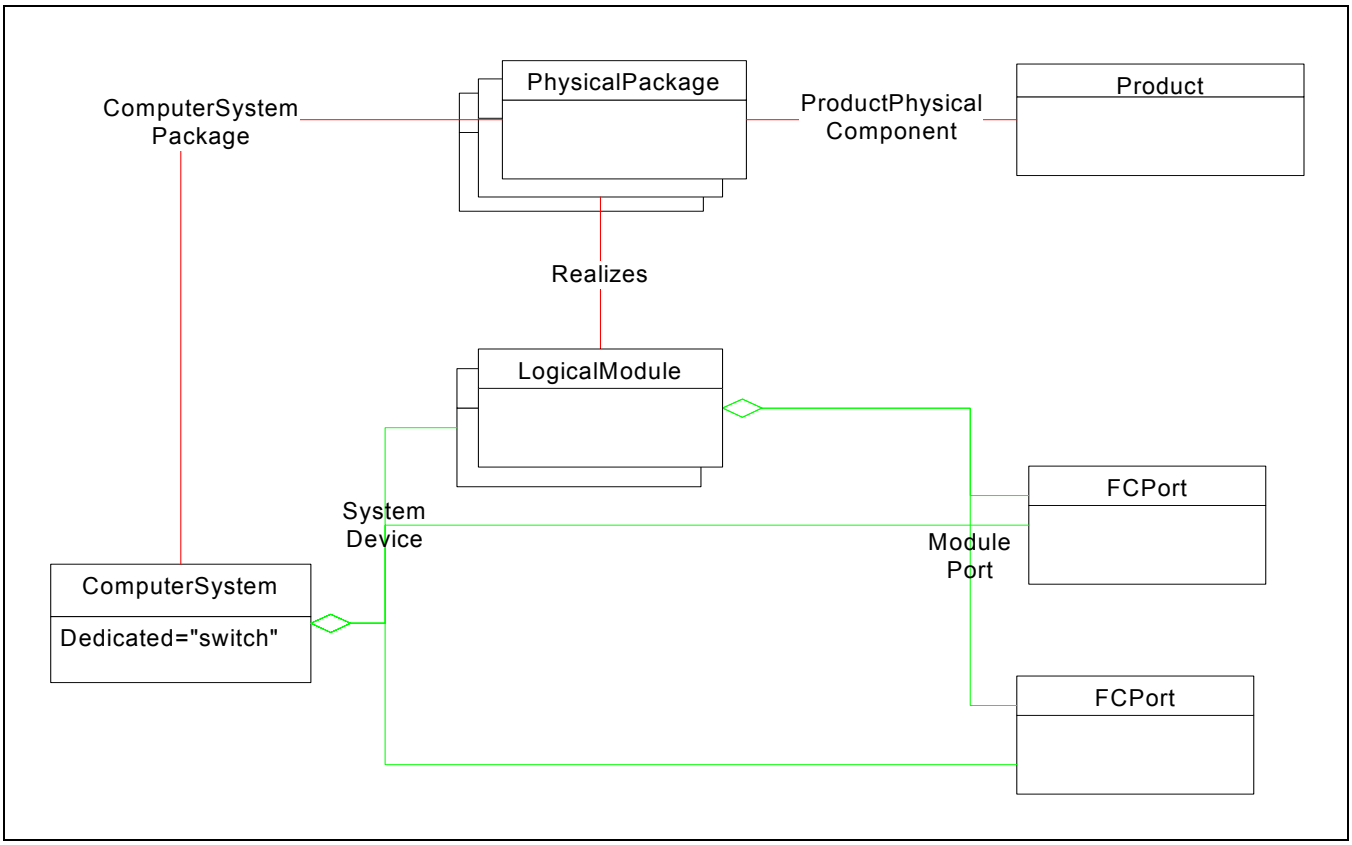

Figure 19 - Switch Blade Instance

14.2 Health and Fault Management

None

14.3 Cascading Considerations

None

14.4 Methods of this Profile

None

14.5 Client Considerations and Recipes

None

14.6 Registered Name and Version

Blades version 1.3.0

14.7 CIM Elements

Table 137 describes the CIM elements for Blades.

Table 137 - CIM Elements for Blades

Element Name	Requirement	Description
14.7.1 CIM_LogicalModule	Mandatory	The Blade
14.7.2 CIM_ModulePort	Mandatory	Associates the LogicalModule to the FCPort
14.7.3 CIM_PhysicalPackage (Logical Module)	Mandatory	The physical package within which the LogicalModule is contained
14.7.4 CIM_Product (Blade)	Optional	The product information for the Blade
14.7.5 CIM_ProductPhysicalComponent	Optional	Associates the Product to the PhysicalPackage
14.7.6 CIM_Realizes (Logical Module Package)	Optional	Associates the LogicalModule to its PhysicalPackage
14.7.7 CIM_SystemDevice (Logical Module)	Mandatory	Associates the LogicalModule to the ComputerSystem representing the Switch
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_LogicalModule	Optional	Creation of an Creation LogicalModule instance.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_LogicalModule	Optional	Deletion of an LogicalModule instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_LogicalModule AND SourceInstance.OperationalStatus <> PreviousInstance.OperationalStatus	Optional	Deprecated WQL -Change in status of LogicalModule.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_LogicalModule AND SourceInstance.CIM_LogicalModule::OperationalStatus <> PreviousInstance.CIM_LogicalModule::OperationalStatus	Optional	CQL -Change in status of LogicalModule.

14.7.1 CIM_LogicalModule

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

Table 138 describes class CIM_LogicalModule.

Table 138 - SMI Referenced Properties/Methods for CIM_LogicalModule

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
DeviceID		Mandatory	Opaque
ElementName		Mandatory	
OperationalStatus		Mandatory	
ModuleNumber		Mandatory	

14.7.2 CIM_ModulePort

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

Table 139 describes class CIM_ModulePort.

Table 139 - SMI Referenced Properties/Methods for CIM_ModulePort

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

14.7.3 CIM_PhysicalPackage (Logical Module)

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

Table 140 describes class CIM_PhysicalPackage (Logical Module).

Table 140 - SMI Referenced Properties/Methods for CIM_PhysicalPackage (Logical Module)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Tag		Mandatory	
ElementName		Optional	
Name		Optional	
Manufacturer		Mandatory	
Model		Mandatory	
SerialNumber		Optional	
Version		Optional	
PartNumber		Optional	

14.7.4 CIM_Product (Blade)

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

Table 141 describes class CIM_Product (Blade).

Table 141 - SMI Referenced Properties/Methods for CIM_Product (Blade)

Properties	Flags	Requirement	Description & Notes
Name		Mandatory	Commonly used Product name.
IdentifyingNumber		Mandatory	Product identification such as a serial number.
Vendor		Mandatory	The manufacturer or the OEM.
Version		Mandatory	Product version information.
ElementName		Mandatory	User Friendly name. Suggested use is Vendor, Version and product name.

14.7.5 CIM_ProductPhysicalComponent

Associates the Product to the PhysicalPackage. This is necessary to link the Product information to the Blade.

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

Table 142 describes class CIM_ProductPhysicalComponent.

Table 142 - SMI Referenced Properties/Methods for CIM_ProductPhysicalComponent

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

14.7.6 CIM_Realizes (Logical Module Package)

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

Table 143 describes class CIM_Realizes (Logical Module Package).

Table 143 - SMI Referenced Properties/Methods for CIM_Realizes (Logical Module Package)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	The reference to the PhysicalPackage.
Dependent		Mandatory	The reference to the LogicalModule representing the Blade.

14.7.7 CIM_SystemDevice (Logical Module)

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

Table 144 describes class CIM_SystemDevice (Logical Module).

Table 144 - SMI Referenced Properties/Methods for CIM_SystemDevice (Logical Module)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

STABLE

EXPERIMENTAL

Clause 15: Switch Partitioning Subprofile

15.1 Description

The Switch Partitioning subprofile is used when a “switch” actually is implementing multiple instances of a profile. The instances of the profile can be the same profiles, for example the Switch profile, or different profiles, for example the Switch Profile and the Extender Profile. For the context of further discussion, the “switch” representing the entire set of systems will be called the Partitioning System and the systems that it is “hosting” are the Partitioned System. For virtual fabrics, ANSI T11 calls the partitioning system the “Core Switch” and the partitioned system the “Virtual Switch”. In Figure 20, an example is shown with a Partitioning System hosting a Partitioned System running the Extender Profile and the Switch Profile. The nomenclature in this profile uses blue to identify the partitioning entity.

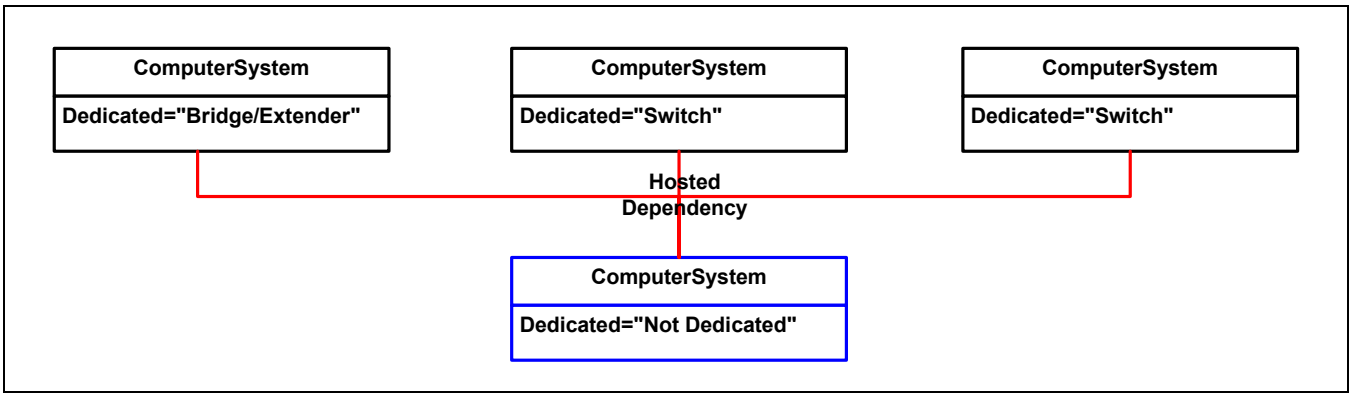

Figure 20 - Switch ComputerSystem and Partitioning System

The relationship between the Partitioning System and its partitions is indicated with the HostedDependency association. The Partitioning System can be determined by locating the ComputerSystem that is not a dependent in a HostedDependency relationship to another ComputerSystem. The Partitioning System also has the Dedicated property set to “Not Dedicated”.

In Figure 21, the Partitioning Ports have been added. For this Profile, the NetworkPort class or a subclass is required to identify the partitioning entity.

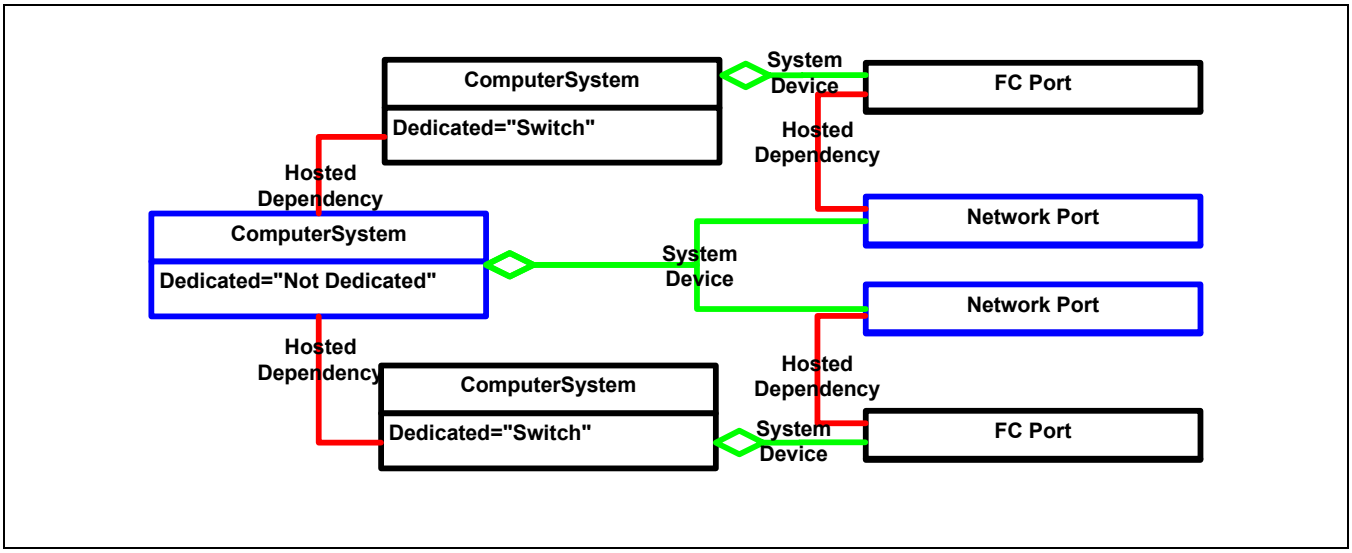

Figure 21 - Switch and Partitioning System and Partitioning Ports

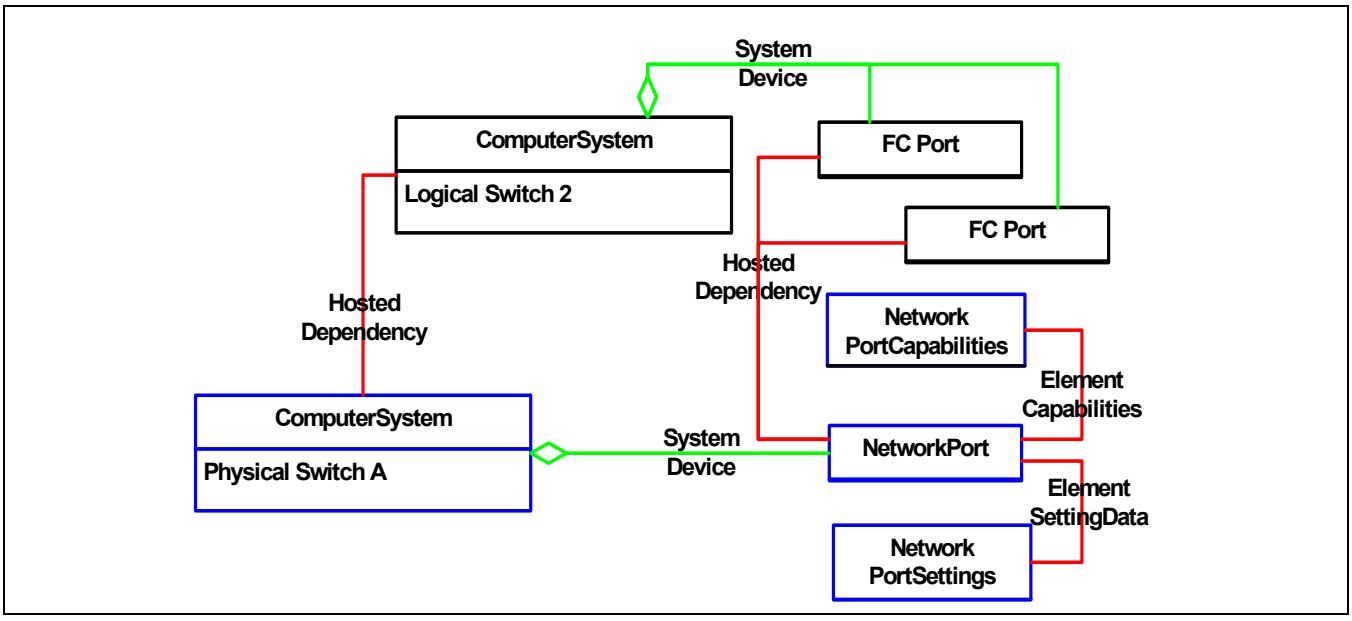

Figure 22 - Underlying System Port Settings and Capabilities

In Figure 22, there is the class NetworkPortCapabilities with the property NetworkIDsConfigurable which indicates whether this port can be assigned to a particular fibre channel fabric. If this property is true, then the property, NetworkIDsFormat should be VF_ID and the array property NetworkIDs can be set to define which fibre channel virtual fabrics this port can belong to. When the port connects to one of the fibre channel fabrics in the setting, an FCPort instance shall be instantiated to represent the Fibre Channel Port that is active in the particular fabric with the association HostedDependency showing which NetworkPort the FCPort was partitioned from. If the setting contained more than one NetworkID, for every active connection to a fabric, an instance of FCPort shall be instantiated.

The FCSwitches partitioned from the partitioning ComputerSystem are implicitly created when a FCPort is connected to a virtual fabric. Currently there is no mechanism to explicitly create a partitioned switch.

15.2 Health and Fault Management Consideration

None

15.3 Cascading Considerations

None

15.4 Supported Profiles, Subprofiles, and Packages

Related Profiles for FabricSwitchPartitioning: Not defined in this standard.

15.5 Methods of the Profile

None

15.6 Client Considerations and Recipes

None

15.7 Registered Name and Version

FabricSwitchPartitioning version 1.2.0

15.8 CIM Elements

Table 145 describes the CIM elements for FabricSwitchPartitioning.

Table 145 - CIM Elements for FabricSwitchPartitioning

Element Name	Requirement	Description
15.8.1 CIM_ComputerSystem (Partitioning)	Mandatory	The partitioning ComputerSystem
15.8.2 CIM_ElementCapabilities (Association between NetworkPort and NetworkPortCapabilities)	Mandatory	Association between NetworkPort and NetworkPortCapabilities.
15.8.3 CIM_ElementSettingData (Association between NetworkPort and NetworkPortSettings)	Mandatory	Association between NetworkPort and NetworkPortSettings
15.8.4 CIM_HostedDependency (NetworkPort to FCPort)	Mandatory	Association between NetworkPort to FCPort
15.8.5 CIM_HostedDependency (Partitioning CS to Partitioned CS)	Mandatory	Association between the Partitioning ComputerSystem and Partitioned ComputerSystem

Table 145 - CIM Elements for FabricSwitchPartitioning

Element Name	Requirement	Description
15.8.6 CIM_NetworkPort	Mandatory	The partitioning port.
15.8.7 CIM_NetworkPortCapabilities	Mandatory	The NetworkPort Capabilities.
15.8.8 CIM_NetworkPortSettings	Mandatory	Defines the Virtual Fabrics the switch port can connect to.
15.8.9 CIM_SystemDevice (NetworkPort to ComputerSystem)	Mandatory	Associates the partitioning classes (NetworkPort to the ComputerSystem)

15.8.1 CIM_ComputerSystem (Partitioning)

The ComputerSystem representing the Interconnect Element (e.g. a switch) or Platform (e.g. Host and Array).

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

Table 146 describes class CIM_ComputerSystem (Partitioning).

Table 146 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Partitioning)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	Name of Class
Name		Mandatory	The Partitioning System Name
Dedicated		Mandatory	

15.8.2 CIM_ElementCapabilities (Association between NetworkPort and NetworkPortCapabilities)

Association between NetworkPort and NetworkPortCapabilities

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

Table 147 describes class CIM_ElementCapabilities (Association between NetworkPort and NetworkPortCapabilities).

Table 147 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (Association between NetworkPort and NetworkPortCapabilities)

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	The reference to the ComputerSystem
Capabilities		Mandatory	The reference to the NetworkPortCapabilities

15.8.3 CIM_ElementSettingData (Association between NetworkPort and NetworkPortSettings)

Association between NetworkPort and NetworkPortSettings

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

Table 148 describes class CIM_ElementSettingData (Association between NetworkPort and NetworkPortSettings).

Table 148 - SMI Referenced Properties/Methods for CIM_ElementSettingData (Association between NetworkPort and NetworkPortSettings)

Properties	Flags	Requirement	Description & Notes
SettingData		Mandatory	The reference to the NetworkPortSettings
ManagedElement		Mandatory	The reference to the ComputerSystem

15.8.4 CIM_HostedDependency (NetworkPort to FCPort)

The association between the partitioning NetworkPort and the partitioned FCPort. The Antecedent references the partitioning port and the Dependent references the partitioned port. The association can be used to determine whether the system is in the underlying SAN topology or the Fabric topology.

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

Table 149 describes class CIM_HostedDependency (NetworkPort to FCPort).

Table 149 - SMI Referenced Properties/Methods for CIM_HostedDependency (NetworkPort to FCPort)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	NetworkPort representing the Partitioning System
Dependent		Mandatory	FCPort representing the Partitioned System

15.8.5 CIM_HostedDependency (Partitioning CS to Partitioned CS)

The association between the Partitioning ComputerSystem and the Partitioned ComputerSystem. The Antecedent references the partitioning system and the Dependent references the partitioned system. The association can be used to determine whether the system is in the underlying SAN topology or the Fabric topology.

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

Table 150 describes class CIM_HostedDependency (Partitioning CS to Partitioned CS).

Table 150 - SMI Referenced Properties/Methods for CIM_HostedDependency (Partitioning CS to Partitioned CS)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	ComputerSystem representing the Partitioning System
Dependent		Mandatory	ComputerSystem representing the Partitioned System

15.8.6 CIM_NetworkPort

The partitioning port.

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

Table 151 describes class CIM_NetworkPort.

Table 151 - SMI Referenced Properties/Methods for CIM_NetworkPort

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	The scoping System's CreationClassName.
SystemName		Mandatory	The scoping System's Name.
CreationClassName		Mandatory	Name of Class
DeviceID		Mandatory	Opaque

15.8.7 CIM_NetworkPortCapabilities

The NetworkPort Capabilities.

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

Table 152 describes class CIM_NetworkPortCapabilities.

Table 152 - SMI Referenced Properties/Methods for CIM_NetworkPortCapabilities

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	Must be set to "NetworkPortCapabilities"

Table 152 - SMI Referenced Properties/Methods for CIM_NetworkPortCapabilities

Properties	Flags	Requirement	Description & Notes
NetworkIDsConfigurable		Mandatory	If the switch supports configuring virtual fabrics, this property mist be TRUE. If the switch only supports discovery, this property must be FALSE.
NetworkIDsFormat		Mandatory	For configuring virtual fabrics, this property must for VF_ID.

15.8.8 CIM_NetworkPortSettings

Defines the Virtual Fabrics the switch port can connect to. The property NetworkIDs is an array which should contain the Virtual Fabric IDs (VF_ID) that the NetworkPort will partition FCPorts for.

Created By: Static

Modified By: Static

Deleted By: Static

Requirement: Mandatory

Table 153 describes class CIM_NetworkPortSettings.

Table 153 - SMI Referenced Properties/Methods for CIM_NetworkPortSettings

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	Must be set to "NetworkPortSettings"
NetworkIDs		Mandatory	Contains the Virtual Fabric IDs that the NetworkPort will host FCPorts for.

15.8.9 CIM_SystemDevice (NetworkPort to ComputerSystem)

Associates the NetworkPort to the ComputerSystem

Created By: External

Modified By: Static

Deleted By: External

Requirement: Mandatory

Table 154 describes class CIM_SystemDevice (NetworkPort to ComputerSystem).

Table 154 - SMI Referenced Properties/Methods for CIM_SystemDevice (NetworkPort to ComputerSystem)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	The reference to the ComputerSystem
PartComponent		Mandatory	The reference to the NetworkPort

EXPERIMENTAL

EXPERIMENTAL

Clause 16: Extender Profile

16.1 Description

A FC Extender is a logical entity representing an inter-switch link consisting of two FC Extender Node devices and the Network pipes that connect them.

A FC Extender is used to connect two Fabrics across a LAN, MAN, WAN, or other network communications media.

A FC Extender Node is a physical device that converts Fibre Channel protocol for transmission over different network communication technologies.

The domain of the Extender Group is defined by Network, which is a subclass of AdminDomain.

16.1.1 FC Extender Node Topology Classes

The ComputerSystem class is the core of the model. It is identified as an Extender node by the dedicated attribute being set to ExtenderNode.

The TCPSettings and IPSettings classes represent the global configuration of the FC Extender transport layer.

The Port group of classes contains the following classes: FCPort, and EthernetPort. The FCPort class represents the connection of a FC Extender to a SAN. This class connects to other FCPort classes to represent Fibre channel

connections. This class could be replaced with other port types to represent SANs based on other interconnect technology. The EthernetPort class represents an Ethernet link between FC Extender nodes.

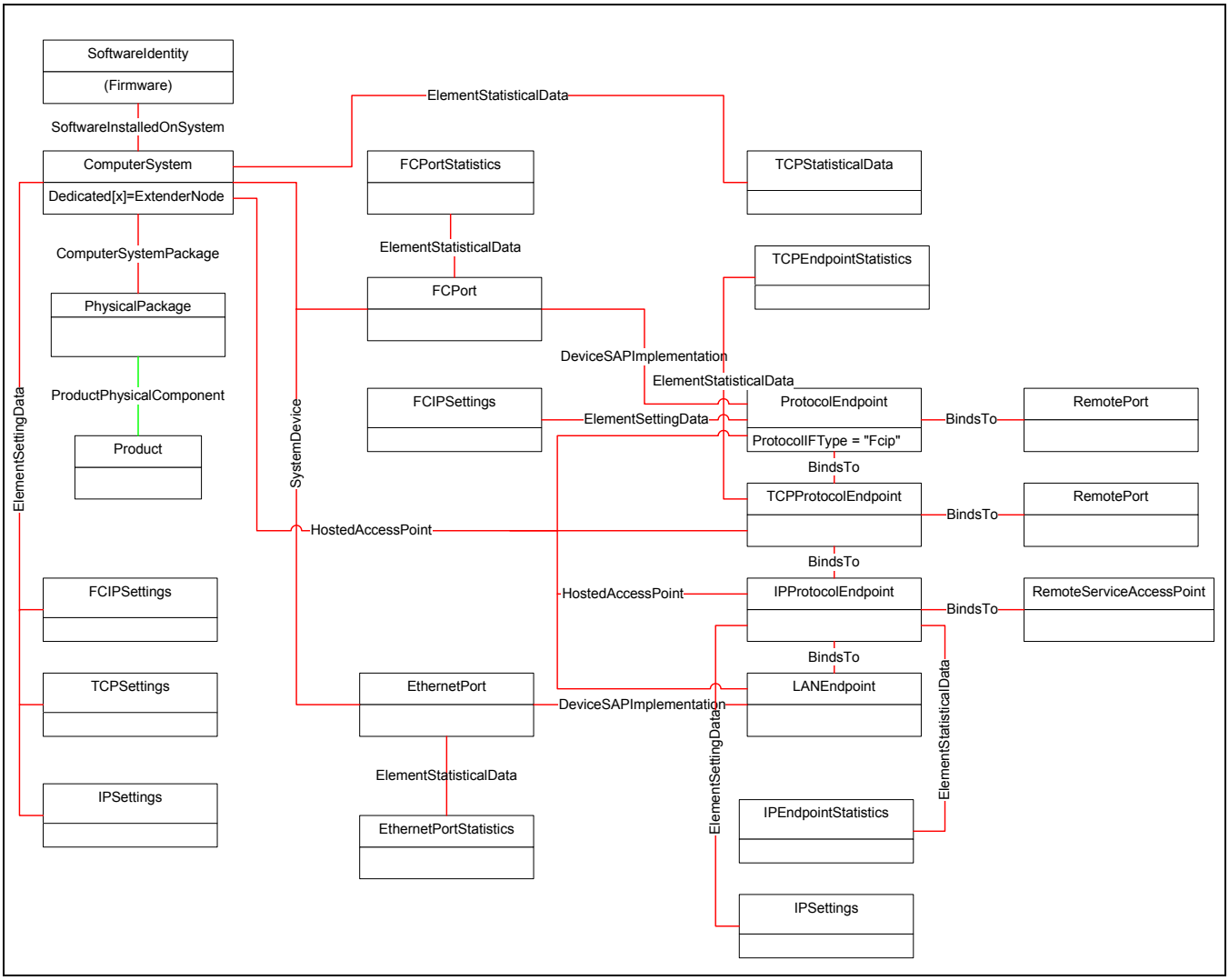

Figure 23 - FC Extender Node Instance

16.1.2 FC Extender Node Network Connectivity Classes

Each FC Extender node local ProtocolEndpoint (e.g., FCProtocolEndpoints, TCPProtocolEndpoints) has a BindsTo dependency on a RemotePort that describes access or addressing information to a remote ProtocolEndpoint for a specific connection.

The Extender node represents ProtocolEndpoints dependencies (e.g., FC ProtocolEndpoint on TCPProtocolEndpoints, TCPProtocolEndpoint on IPProtocolEndpoint, IPProtocolEndpoint on EthernetProtocolEndpoint) with a BindsTo association.

16.1.3 FC Extender Group Network Connectivity Classes

A FC Extender connection is represented by a NetworkPipe class associated with FCProtocolEndpoints. A FCExtender Network class groups multiple NetworkPipes.

The NetworkPipe between FCProtocolEndpoints is composed of lower-level TCP network pipes.

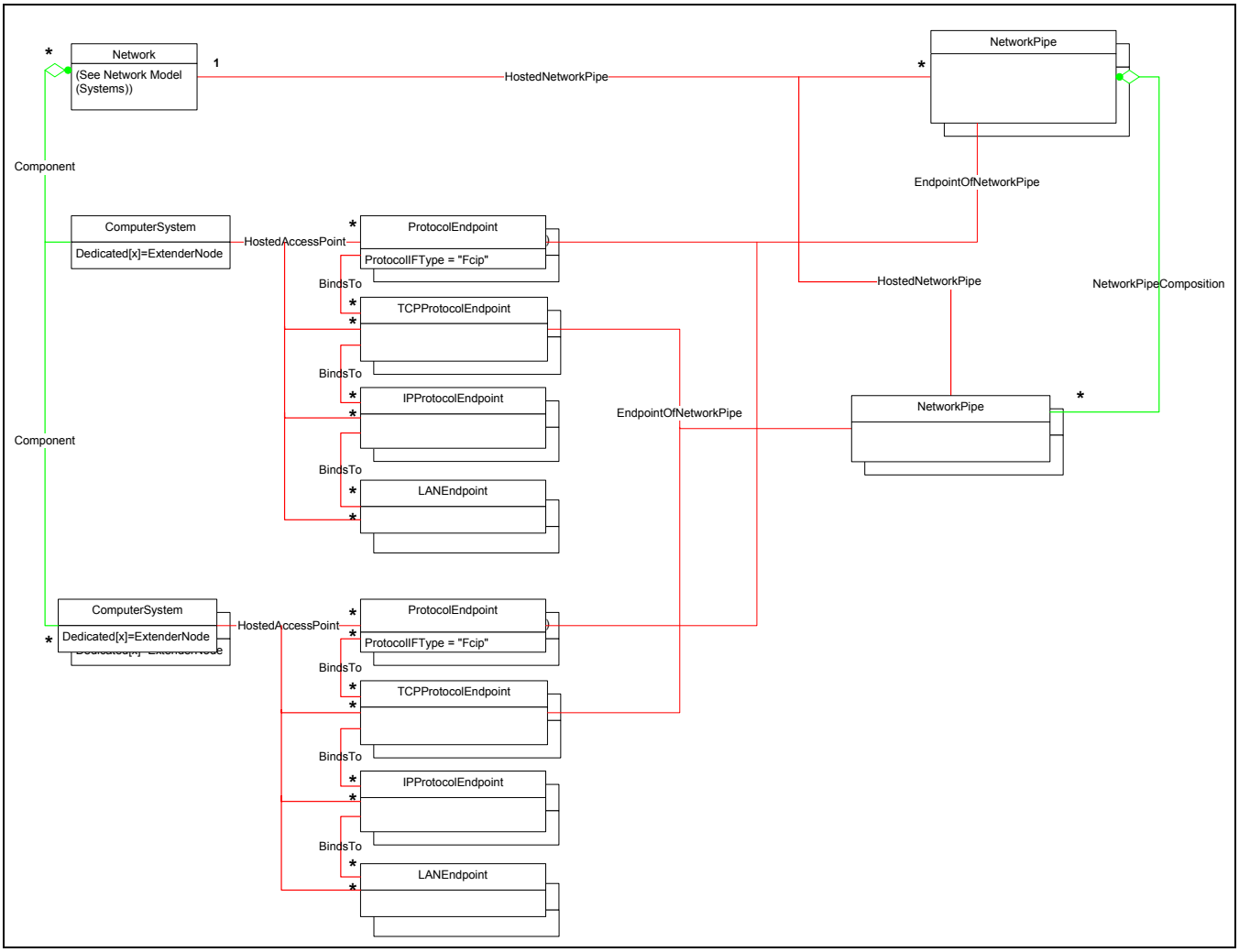

Figure 24 - FC Extender Group Instance

16.2 Health and Fault Management

None

16.3 Cascading Considerations

None

16.4 Supported Subprofiles and Packages

None

16.5 Methods of this Profile

None

16.6 Client Considerations and Recipes

16.6.1 Extender Connectivity Settings

```
// Description
// Collecting settings
// of Extender node connectivity elements participating in the Extender
// of interest.
// $extenderNodeFCIPSettings(fcip protocol endpoint settings)
// $extenderNodeTCPSettings (transport layer settings)
// $extenderIPSettings (ip protocol endpoint settings)
// PREEXISTING CONDITIONS AND ASSUMPTIONS
// The Extender fcip ProtocolEndpoint has been previously
// identified and defined in the $fcipProtocolEndpoint-> variable
// 1. Get ComputerSystem associated with $fcipProtocolEndpoint

$extenderNodes[] = Associators(
     $fcipProtocolEndpoint->,
     CIM_HostedAccessPoint,
     CIM_ComputerSystem,
     Antecedent,
     Dependent,
     false,
     false,
     [Dedicated])
if (contains(23, $extenderNodes[0].Dedicated))
{
     #extenderNodeAccessA = true
     $extenderNode-> = $extenderNodes[0].getObjectPath()
}
if(#extenderNodeAccessA)
{
//2. Get fcip protocol endpoint

 // find FCIP Settings
     $fcipSettings[]= Associators(
        $fcipProtocolEndpoint,
        CIM_ElementSettingData,
        CIM_FCIPSettings,
        ManagedElement,
        SettingData,
        false,
        false,
        null)
     if ($fcipSettings[].length != 0)
        $extenderNodeFCIPSettings = $fcipSettings[0]
```

```
//3. Get transport layer settings

     $tcpSettings[]= Associators(
        $extenderNode->,
        CIM_ElementSettingData,
        CIM_TCPSettings,
        ManagedElement,
        SettingData,
        false,
        false,
        null)
     if ($tcpSettings[].length != 0)
        $extenderNodeTCPSettings = $tcpSettings[0]


//4. Find TCPProtocolEndpoint bound to the extender fcip ProtocolEndpoint


     $tcpProtocolEndpoint->[]= AssociatorNames(
        $fcipProtocolEndpoint->,
        CIM_BindTo,
        CIM_TCPProtocolEndpoint,
        Dependent,
        Antecedent,
        false,
        false,
        null))

//5. Find IPProtocolEndpoint bound to the extender tcp ProtocolEndpoint


     $ipProtocolEndpoints->[]= AssociatorNames(
        $tcpProtocolEndpoint->,
        CIM_BindTo,
        CIM_IPProtocolEndpoint,
        Dependent,
        Antecedent,
        false,
        false,
        null))
        $ipProtocolEndpoint-> = $ipProtocolEndpoints->[0]

//6. Find IPProtocolEndpoint settings


     $ipSettings[]= Associators(
        $ipProtocolEndpoint->,
```

```
            CIM_ElementSettingData,
            CIM_IPSettings,
            ManagedElement,
            SettingData,
            false,
            false,
            null)
        if ($ipSettings[].length != 0)
            $extenderNodeIPSettings = $ipSettings[0]


}
```

16.6.2 Extender Connective Statistics

```
// Description
// Collecting statistical data
// of Extender node conectivity elements participating in the Extender
// of interest.
// $extenderNodeTCPStatisticalData (transport layer stats)
// $extenderIPEndpointStatistics (IP protocol endpoint stats)
// PREEXISTING CONDITIONS AND ASSUMPTIONS
// The Extender fcip ProtocolEndpoint has been previously
// identified and defined in the $fcipProtocolEndpoint-> variable
// 1. Get ComputerSystem associated with $fcipProtocolEndpoint

$extenderNodes[] = Associators(
     $fcipProtocolEndpoint->,
     CIM_HostedAccessPoint,
     CIM_ComputerSystem,
     Antecedent,
     Dependent,
     false,
     false,
     [Dedicated])
if (contains(23, $extenderNodes[0].Dedicated))
{
     #extenderNodeAccess = true
     $extenderNode-> = $extenderNodes[0].getObjectPath()
}
if(#extenderNodeAccess)
{
//2. Get transport layer statistics
```

```
        $tcpStatistics[] = Associators(
               $extenderNode->,
               CIM_ElementStatisticalData,
               CIM_TCPStatisticalData,
               ManagedElement,
               Stats,
               false,
               false,
               null))
        $extenderNodeTCPStatisticalData = $tcpStatistics[0]

    //3. Find TCPProtocolEndpoint bound to the extender fcip ProtocolEndpoint


        $tcpProtocolEndpoint->[]= AssociatorNames(
           $fcipProtocolEndpoint->,
           CIM_BindTo,
           CIM_TCPProtocolEndpoint,
           Dependent,
           Antecedent,
           false,
           false,
           null))

    //4. Find IPProtocolEndpoint bound to the extender tcp ProtocolEndpoint


        $ipProtocolEndpoint->[]= AssociatorNames(
           $tcpProtocolEndpoint->,
           CIM_BindTo,
           CIM_IPProtocolEndpoint,
           Dependent,
           Antecedent,
           false,
           false,
           null))

    //5. Find IPProtocolEndpoint statistics


        $ipStatistics[]= Associators(
           $ipProtocolEndpoint->,
           CIM_ElementStatisticalData,
           CIM_IPEndpointStatistics,
           ManagedElement,
           Stats,
           false,
```

```
        false,
        null)

    $extenderIPEndpointStatistics = $ipStatistics[0]

}
```

16.6.3 Extender Port Group Information

```
// Description
// Collecting configuration and statistical data
// of Extender node ports participating in the Extender
// of interest.
// $extenderNodeFCPort (connected to a switch)
// $extenderNodeFCPortStatistics
// $extenderNodeEthernetPort (connected to a peer Extender node)
// $extenderNodeEthernetStatistics
// PREEXISTING CONDITIONS AND ASSUMPTIONS
// The Extender fcip ProtocolEndpoint has been previously
// identified and defined in the $fcipProtocolEndpoint-> variable

// 1. Get ComputerSystem associated with $fcipProtocolEndpoint

$extenderNodes[] = Associators(
     $fcipProtocolEndpoint->,
     CIM_HostedAccessPoint,
     CIM_ComputerSystem,
     Antecedent,
     Dependent,
     false,
     false,
     [Dedicated])
if (contains(23, $extenderNodes[0].Dedicated))
{
     #extenderNodeAccess = true
}
if(#extenderNodeAccess)
{
     // 2. Get FC port

     $fcPorts[] = Associators(
        $fcipProtocolEndpoints->,
        CIM_DeviceSAPImplementation,
        CIM_FCPort,
        Dependent,
        Antecedent,
        false,
```

```
        false,
        null)
    $extenderNodeFCPort = $fcPorts[0]

    // 2. Get FC port statistics

    $fcPortStatistics->[] = Associators(
        $extenderNodeFCPort.getObjectPath(),
        CIM_ElementStatisticalData,
        CIM_FCPortStatistics,
        ManagedElement,
        Stats,
        false,
        false,
        null))
        $extenderNodeFCPortStatistics = $fcPortsStatistics[0]

    //3. Find TCPProtocolEndpoint bound to the extender FCIP ProtocolEndpoint

    $tcpProtocolEndpoints->[]= AssociatorNames(
        $fcipProtocolEndpoint->,
        CIM_BindTo,
        CIM_TCPProtocolEndpoint,
        Dependent,
        Antecedent,
        false,
        false,
        null))
    // at least one should exist
    $tcpProtocolEndpoint->=$tcpProtocolEndpoints->[0]


//4. Find IPProtocolEndpoint bound to the extender TCP ProtocolEndpoint

    $ipProtocolEndpoint->[]= AssociatorNames(
        $tcpProtocolEndpoint->,
        CIM_BindTo,
        CIM_IPProtocolEndpoint,
        Dependent,
        Antecedent,
        false,
        false,
        null))
    $ipProtocolEndpoint->=$ipProtocolEndpoints->[0]

//5. Get Ethernet port
```

```
      $ethernetPorts[] = Associators(
         $ipProtocolEndpoints->,
         CIM_DeviceSAPImplementation,
         CIM_EthernetPort,
         Dependent,
         Antecedent,
         false,
         false,
         null)
      $extenderNodeEthernetPort = $ethernetPorts[0]

//6. Get Ethernet port statistics

      $ethernetPortStatistics->[] = Associators(
         $extenderNodeEthernetPort.getObjectPath(),
         CIM_ElementStatisticalData,
         CIM_EthernetStatistics,
         ManagedElement,
         Stats,
         false,
         false,
         null))
      $extenderNodeEthernetPortStatistics = $ethernetPortsStatistics[0]


}
```

16.6.4 Extender Topology Mapping

```
// This recipe describes how to build a topology graph of a fabric.
//
// 1. Identifies all the Switches and adds their objects paths and the
// object paths of the FC Ports belonging to these Switches to the $nodes
// array
//
// 2. Creates a suitable Association instance (e.g. a SystemDevice
// Association instance between a Switch and a FC Port), setting its
// GroupComponent and PartComponent. Adds the object path of the
// Association to the $links array
//
// 3. Creates a map of all connected FC Ports (i.e., belonging to Switches
// that are ISL'd together and to Host HBAs and Storage System Front End
// Controllers)
//
```

```
// In this map, the FC Ports (i.e., the ones that are connected) are
// cross-connected.
//
// e.g., For a pair of FC Ports, one belonging to a Switch and the other
// belonging to a Host (HBA), the map indexed by the Switch Port WWN returns
// the Host (HBA) FC Port object path and the map indexed by the Host (HBA)
// FC Port WWN returns the Switch FC Port object path.
//
// Similar relationship exists between the pairs of FC Ports where one
// belongs to a Switch and the other belonging belongs to a Storage System
// Front End Controller and for FC Ports each of which belongs to a Switch.
//
// 4. Identifies all the Hosts and adds their objects paths to the $nodes
// array. Note that the object paths of the FC Ports (HBA Ports) belonging
// to these Hosts are already added to the $nodes array in step-3.
//
// 5. Creates a suitable Association instance (e.g. a SystemDevice
// Association instance between a Host and a FC Port), setting its
// GroupComponent and PartComponent. Adds the object path of the Association
// to the $links array.
//
// 6. Identifies all the Storage Systems and adds their objects paths to the
// $nodes array.
// Note that the object paths of the FC Ports (i.e., Front End Controller
// FC Ports) belonging to these Storage Systems are already added to the
// $nodes array in step-3.
//
// 7. Creates a suitable Association instance (e.g. a SystemDevice
// Association instance between a Storage System and a FC Port), setting
// its GroupComponent and PartComponent. Adds the object path of the
// Association to the $links array.

// DESCRIPTION
// Create a map of how elements in a SAN are connected together via
// Fibre-ChannelFC ports.
//
// The map is built in array $attachedFcPorts->[], where the index is a
// WWN of any device port on the SAN, and the value at that index is
// the object path of the connected Switch or HBA or Storage System FC port.
//
// First find all the switches in a SAN. Get all the FC Ports for each
// switch and get the Attached FC Ports for each Switch FC Port. Save these
// device FC ports in the map described above.

// PREEXISTING CONDITIONS AND ASSUMPTIONS
// 1. All agents/namespaces supporting Fabric Profile previously identified
// using SLP. Do this for each CIMOM supporting Fabric Profile
```

```
switches[] = enumerateInstances("CIM_ComputerSystem", true, false, true, true,
                            null)
for #i in $switches[]
{
    if (!contains(5, $switches[#i].Dedicated))
        continue

    // only process switches, not other computer systems

    // Add the switch to the $nodes array

    $nodes.addIfNotAlreadyAdded ($switches[#i].getObjectPath();

    // Get all the SystemDevice associations between this switch and its
    // FC Ports

    $sysDevAssoc[] = ReferenceNames($switches[#i],
                                "CIM_FCPort",
                                "GroupComponent");

    // Add these associations to the $links array

    for #a in $sysDevAssoc->[]
    $links.addIfNotAlreadyAdded ($sysDevAssoc->[#a];

    $fcPorts->[] = AssociatorNames(
        $switches[#i].getObjectPath(),
        "CIM_SystemDevice",
        "CIM_FCPort",
        "GroupComponent",
        "PartComponent")
    for #j in $fcPorts->[]
    {

        // Add the FC Port in $nodes array

        $nodes.addIfNotAlreadyAdded (fcPorts->[#j];

        $protocolEndpoints->[] = AssociatorNames(
            fcPorts->[#j],
            "CIM_DeviceSAPImplementation",
            "CIM_ProtocolEndpoint",
            "Antecedent",
        "Dependent");

        // NOTE - It is possible for this collection to be empty (i.e., ports
        // that are not connected). It is possible for this collection to
```

```
// have more than one element (loops attached to a switch port is the
// most common example).

if ($protocolEndpoints->[].length == 0)
    continue

// Add the Protocol End Point to the nodes array.
// Currently this recipe is designed to only save one
// ProtocolEndpoint.

$nodes.addIfNotAlreadyAdded (protocolEndpoints[0]);

// Add the associations between the fcPort and the Protocol end point
// to the links array

$devSAPImplassoc[]  = ReferenceNames($fcPorts->[#j],
                              “CIM_ProtocolEndpoint”,
                              null);
for #a in $devSAPImplassoc->[]
    $links.addIfNotAlreadyAdded ($devSAPImplassoc->[#a];

$attachedProtocolEndpoints->[] = AssociatorNames(
    $protocolEndpoints->[0],
    “CIM_ActiveConnection”,
    “CIM_ProtocolEndpoint”,
    null, null)

// Add the Attached Protocol End Point to the nodes array

$nodes.addIfNotAlreadyAdded (attachedProtocolEndpoints->[0]);

// Add the associations between the Protocol end point and the
// Attached protocol endpoint to the links array

$actConnassoc[]  = ReferenceNames($protocolEndpoint->[#0],
                           “CIM_ActiveConnection”,
                            null);
for #a in $actConnassoc->[]
    $links.addIfNotAlreadyAdded ($actConnassoc->[#a];

// NOTE: role & resultRole are null as the direction of the
// association is not dictated by the specification

// $attachedFcPort is either a device FC port or an ISL’d switch FC
// port from another switch. We store this result is stored (i.e.
// which device FC Port is connected // to which switch FC Port) in
// a suitable data structure for subsequent correlation to ports
```

```
            // discovered on devices.

            for #k in $attachedProtocolEndpoints->[]
            {
                $attachedFcPorts->[] = Associators(
                    $attachedProtocolEndpoints->[#k],
                    "CIM_DeviceSAPImplementation",
                    "CIM_FCPort",
                    "Dependent",
                    "Antecedent",
                    false,
                    false,
                    ["PermanentAddress"])
               $attachedFcPort = $attachedFcPorts[0] // Exactly one member guaranteed
                               by model

                // Add the attached FC Port to the $nodes array
                if $attachedFcPort != null
                    $nodes.addIfNotAlreadyAdded ($attachedFcPort);
            }
        }
    }
```

16.7 Registered Name and Version

Extender version 1.2.0

16.8 CIM Elements

Table 155 describes the CIM elements for Extender.

Table 155 - CIM Elements for Extender

Element Name	Requirement	Description
16.8.1 CIM_BindsTo (IPPE to PE)	Mandatory	Associates Extender Node ProtocolEndpoints from different layers in the protocol stack
16.8.2 CIM_BindsTo (IPPE to REmoteSAP)	Mandatory	Associates Extender Node ProtocolEndpoints from different layers in the protocol stack
16.8.3 CIM_BindsTo (IPPE to RemoteSAP)	Mandatory	Associates Extender Node ProtocolEndpoints from different layers in the protocol stack
16.8.4 CIM_BindsTo (PE to RemotePort)	Mandatory	Associates Extender Node ProtocolEndpoints from different layers in the protocol stack
16.8.5 CIM_BindsTo (TCPPE to IPPE)	Mandatory	Associates Extender Node ProtocolEndpoints from different layers in the protocol stack

Table 155 - CIM Elements for Extender

Element Name	Requirement	Description
16.8.6 CIM_BindsTo (TCPPE to PE)	Mandatory	Associates Extender Node ProtocolEndpoints from different layers in the protocol stack
16.8.7 CIM_BindsTo (TCPPE to RemotePort)	Mandatory	Associates Extender Node ProtocolEndpoints from different layers in the protocol stack
16.8.8 CIM_Component	Mandatory	Aggregates Extender Nodes in the Network that represents the group of Extenders
16.8.9 CIM_ComputerSystem	Mandatory	Represents the Extender Node
16.8.10 CIM_DeviceSAPImplementation	Mandatory	
16.8.11 CIM_ElementSettingData (IPPE to IPSettings)	Mandatory	Associates SettingData to Extender Node or ProtocolEndpoints
16.8.12 CIM_ElementSettingData (PE to FCIPSettings)	Mandatory	Associates SettingData to Extender Node or ProtocolEndpoints
16.8.13 CIM_ElementSettingData (System to FCIPSettings)	Mandatory	Associates SettingData to Extender Node or ProtocolEndpoints
16.8.14 CIM_ElementSettingData (System to IPSettings)	Mandatory	Associates SettingData to Extender Node or ProtocolEndpoints
16.8.15 CIM_ElementSettingData (System to TCPSettings)	Mandatory	Associates SettingData to Extender Node or ProtocolEndpoints
16.8.16 CIM_ElementSettingData (TCPPE to TCPSettings)	Mandatory	Associates SettingData to Extender Node or ProtocolEndpoints
16.8.17 CIM_ElementStatisticalData (EthernetPort to EthernetPortStatistics)	Mandatory	Associates StatisticalData to Extender Node or ProtocolEndpoints
16.8.18 CIM_ElementStatisticalData (FCPort to FCPortStatistics)	Mandatory	
16.8.19 CIM_ElementStatisticalData (IPPE to IPEndpointStatistics)	Mandatory	
16.8.20 CIM_ElementStatisticalData (System to TCPStatisticalData)	Mandatory	
16.8.21 CIM_ElementStatisticalData (TCPPE to TCPEndpointStatistics)	Mandatory	
16.8.22 CIM_EndpointOfNetworkPipe (PE to NetworkPipe)	Mandatory	
16.8.23 CIM_EndpointOfNetworkPipe (TCPPE to NetworkPipe)	Mandatory	
16.8.24 CIM_EthernetPort	Mandatory	
16.8.25 CIM_EthernetPortStatistics	Mandatory	

Table 155 - CIM Elements for Extender

Element Name	Requirement	Description
16.8.26 CIM_FCIPSettings	Mandatory	Defines FCIP settings for a group of ProtocolEndpoints (ProtocolIFType - "Fcip") which belongs to the ComputerSystem (Extender Node)
16.8.27 CIM_FCPort	Mandatory	
16.8.28 CIM_FCPortStatistics	Mandatory	
16.8.29 CIM_HostedAccessPoint (ComputerSYstem to IPProtocolEndpoint)	Mandatory	Associates the ProtocolEndpoint to the ComputerSystem or Network
16.8.30 CIM_HostedAccessPoint (ComputerSYstem to TCPProtocolEndpoint)	Mandatory	Associates the ProtocolEndpoint to the ComputerSystem or Network
16.8.31 CIM_HostedAccessPoint (ComputerSystem to ProtocolEndpoint)	Mandatory	Associates the ProtocolEndpoint to the ComputerSystem or Network
16.8.32 CIM_HostedNetworkPipe	Mandatory	Associates NetworkPipe to the Network
16.8.33 CIM_IPEndpointStatistics	Mandatory	
16.8.34 CIM_IPProtocolEndpoint	Mandatory	
16.8.35 CIM_IPSettings	Mandatory	Defines IP settings for a group of IPProtocolEndpoints which belongs to the ComputerSystem
16.8.36 CIM_LANEndpoint	Mandatory	The ProtocolEndpoint for the Ethernet port.
16.8.37 CIM_Network	Mandatory	Network represents a network connectivity domain. It groups NetworkPipes.
16.8.38 CIM_NetworkPipe	Mandatory	NetworkPipe represents state, configuration of a connection between endpoints in the context of a Network
16.8.39 CIM_NetworkPipeComposition	Mandatory	
16.8.40 CIM_ProtocolEndpoint	Mandatory	The ProtocolEndpoint representing the FCIP layer. ProtocolEndpoint shall be implemented when an ActiveConnection or NetworkPipe exists. It may be implemented if no ActiveConnection or NetworkPipe exists.
16.8.41 CIM_RemotePort	Optional	
16.8.42 CIM_RemoteServiceAccessPoint	Optional	
16.8.43 CIM_SystemDevice (System to EthernetPort)	Mandatory	Associated FCPort and EthernetPort to the ComputerSystem
16.8.44 CIM_SystemDevice (System to FCPort)	Mandatory	Associated FCPort and EthernetPort to the ComputerSystem
16.8.45 CIM_TCPEndpointStatistics	Mandatory	Opaque
16.8.46 CIM_TCPProtocolEndpoint	Mandatory	

Table 155 - CIM Elements for Extender

Element Name	Requirement	Description
16.8.47 CIM_TCPSettings	Mandatory	Defines TCP settings for a group of TCPProtocolEndpoints which belongs to the ComputerSystem
16.8.48 CIM_TCPStatisticalData	Mandatory	
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_ComputerSystem	Mandatory	Creation of a ComputerSystem instance
SELECT * FROM CIM_InstDeletion WHERE SourceInstance CIM_ComputerSystem	Mandatory	Deletion of a computer system instance
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND SourceInstance.Operationalstatus ** PreviousInstance.Operationalstatus	Mandatory	Deprecated WQL - Change of OperationalStatus for a Computer System
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND SourceInstance.CIM_ComputerSystem::Operationalstatus ** PreviousInstance.CIM_ComputerSystem::Operationalstatus	Mandatory	CQL -Change of OperationalStatus for a Computer System
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND SourceInstance.Operationalstatus ** PreviousInstance.Operationalstatus	Mandatory	Deprecated WQL - Change of OperationalStatus for a Computer System
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND SourceInstance.CIM_ComputerSystem::Operationalstatus ** PreviousInstance.CIM_ComputerSystem::Operationalstatus	Mandatory	CQL -Change of OperationalStatus for a Computer System

16.8.1 CIM_BindsTo (IPPE to PE)

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

Table 156 describes class CIM_BindsTo (IPPE to PE).

Table 156 - SMI Referenced Properties/Methods for CIM_BindsTo (IPPE to PE)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

16.8.2 CIM_BindsTo (IPPE to REmoteSAP)

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

Table 157 describes class CIM_BindsTo (IPPE to REmoteSAP).

Table 157 - SMI Referenced Properties/Methods for CIM_BindsTo (IPPE to REmoteSAP)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

16.8.3 CIM_BindsTo (IPPE to RemoteSAP)

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

Table 158 describes class CIM_BindsTo (IPPE to RemoteSAP).

Table 158 - SMI Referenced Properties/Methods for CIM_BindsTo (IPPE to RemoteSAP)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

16.8.4 CIM_BindsTo (PE to RemotePort)

Created By: Static
Modified By: Static
Deleted By: Static

Requirement: Mandatory

Table 159 describes class CIM_BindsTo (PE to RemotePort).

Table 159 - SMI Referenced Properties/Methods for CIM_BindsTo (PE to RemotePort)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

16.8.5 CIM_BindsTo (TCPPE to IPPE)

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

Table 160 describes class CIM_BindsTo (TCPPE to IPPE).

Table 160 - SMI Referenced Properties/Methods for CIM_BindsTo (TCPPE to IPPE)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

16.8.6 CIM_BindsTo (TCPPE to PE)

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

Table 161 describes class CIM_BindsTo (TCPPE to PE).

Table 161 - SMI Referenced Properties/Methods for CIM_BindsTo (TCPPE to PE)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

16.8.7 CIM_BindsTo (TCPPE to RemotePort)

Created By: Static

Modified By: Static
Deleted By: Static
Requirement: Mandatory

Table 162 describes class CIM_BindsTo (TCPPE to RemotePort).

Table 162 - SMI Referenced Properties/Methods for CIM_BindsTo (TCPPE to RemotePort)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

16.8.8 CIM_Component

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

Table 163 describes class CIM_Component.

Table 163 - SMI Referenced Properties/Methods for CIM_Component

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

16.8.9 CIM_ComputerSystem

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

Table 164 describes class CIM_ComputerSystem.

Table 164 - SMI Referenced Properties/Methods for CIM_ComputerSystem

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Name		Mandatory	IP Address
ElementName		Optional	User friendly name
NameFormat		Mandatory	IP Address

Table 164 - SMI Referenced Properties/Methods for CIM_ComputerSystem

Properties	Flags	Requirement	Description & Notes
OtherIdentifyingInfo		Optional	DNS name
OperationalStatus		Mandatory	Status of Computer System.
IdentifyingDescriptions		Optional	Fully qualified domain name
Dedicated		Mandatory	ExtenderNode

16.8.10 CIM_DeviceSAPImplementation

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

Table 165 describes class CIM_DeviceSAPImplementation.

Table 165 - SMI Referenced Properties/Methods for CIM_DeviceSAPImplementation

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	
Dependent		Mandatory	
Antecedent		Mandatory	

16.8.11 CIM_ElementSettingData (IPPE to IPSettings)

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

Table 166 describes class CIM_ElementSettingData (IPPE to IPSettings).

Table 166 - SMI Referenced Properties/Methods for CIM_ElementSettingData (IPPE to IPSettings)

Properties	Flags	Requirement	Description & Notes
IsDefault		Mandatory	
IsCurrent		Mandatory	

Table 166 - SMI Referenced Properties/Methods for CIM_ElementSettingData (IPPE to IPSettings)

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	
SettingData		Mandatory	

16.8.12 CIM_ElementSettingData (PE to FCIPSettings)

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

Table 167 describes class CIM_ElementSettingData (PE to FCIPSettings).

Table 167 - SMI Referenced Properties/Methods for CIM_ElementSettingData (PE to FCIPSettings)

Properties	Flags	Requirement	Description & Notes
IsDefault		Mandatory	
IsCurrent		Mandatory	
SettingData		Mandatory	
ManagedElement		Mandatory	

16.8.13 CIM_ElementSettingData (System to FCIPSettings)

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

Table 168 describes class CIM_ElementSettingData (System to FCIPSettings).

Table 168 - SMI Referenced Properties/Methods for CIM_ElementSettingData (System to FCIPSettings)

Properties	Flags	Requirement	Description & Notes
IsDefault		Mandatory	
IsCurrent		Mandatory	
SettingData		Mandatory	
ManagedElement		Mandatory	

16.8.14 CIM_ElementSettingData (System to IPSettings)

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

Table 169 describes class CIM_ElementSettingData (System to IPSettings).

Table 169 - SMI Referenced Properties/Methods for CIM_ElementSettingData (System to IPSettings)

Properties	Flags	Requirement	Description & Notes
IsDefault		Mandatory	
IsCurrent		Mandatory	
SettingData		Mandatory	
ManagedElement		Mandatory	

16.8.15 CIM_ElementSettingData (System to TCPSettings)

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

Table 170 describes class CIM_ElementSettingData (System to TCPSettings).

Table 170 - SMI Referenced Properties/Methods for CIM_ElementSettingData (System to TCPSettings)

Properties	Flags	Requirement	Description & Notes
IsDefault		Mandatory	
IsCurrent		Mandatory	
SettingData		Mandatory	
ManagedElement		Mandatory	

16.8.16 CIM_ElementSettingData (TCPPE to TCPSettings)

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

Table 171 describes class CIM_ElementSettingData (TCPPE to TCPSettings).

Table 171 - SMI Referenced Properties/Methods for CIM_ElementSettingData (TCPPE to TCPSettings)

Properties	Flags	Requirement	Description & Notes
IsDefault		Mandatory	
IsCurrent		Mandatory	
SettingData		Mandatory	
ManagedElement		Mandatory	

16.8.17 CIM_ElementStatisticalData (EthernetPort to EthernetPortStatistics)

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

Table 172 describes class CIM_ElementStatisticalData (EthernetPort to EthernetPortStatistics).

Table 172 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (EthernetPort to EthernetPortStatistics)

Properties	Flags	Requirement	Description & Notes
Stats		Mandatory	
ManagedElement		Mandatory	

16.8.18 CIM_ElementStatisticalData (FCPort to FCPortStatistics)

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

Table 173 describes class CIM_ElementStatisticalData (FCPort to FCPortStatistics).

Table 173 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (FCPort to FCPortStatistics)

Properties	Flags	Requirement	Description & Notes
Stats		Mandatory	
ManagedElement		Mandatory	

16.8.19 CIM_ElementStatisticalData (IPPE to IPEndpointStatistics)

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

Table 174 describes class CIM_ElementStatisticalData (IPPE to IPEndpointStatistics).

Table 174 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (IPPE to IPEndpointStatistics)

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	
Stats		Mandatory	

16.8.20 CIM_ElementStatisticalData (System to TCPStatisticalData)

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

Table 175 describes class CIM_ElementStatisticalData (System to TCPStatisticalData).

Table 175 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (System to TCPStatisticalData)

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	
Stats		Mandatory	

16.8.21 CIM_ElementStatisticalData (TCPPE to TCPEndpointStatistics)

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

Table 176 describes class CIM_ElementStatisticalData (TCPPE to TCPEndpointStatistics).

Table 176 - SMI Referenced Properties/Methods for CIM_ElementStatisticalData (TCPPE to TCPEndpointStatistics)

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	
Stats		Mandatory	

16.8.22 CIM_EndpointOfNetworkPipe (PE to NetworkPipe)

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

Table 177 describes class CIM_EndpointOfNetworkPipe (PE to NetworkPipe).

Table 177 - SMI Referenced Properties/Methods for CIM_EndpointOfNetworkPipe (PE to NetworkPipe)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

16.8.23 CIM_EndpointOfNetworkPipe (TCPPE to NetworkPipe)

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

Table 178 describes class CIM_EndpointOfNetworkPipe (TCPPE to NetworkPipe).

Table 178 - SMI Referenced Properties/Methods for CIM_EndpointOfNetworkPipe (TCPPE to NetworkPipe)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

16.8.24 CIM_EthernetPort

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

Table 179 describes class CIM_EthernetPort.

Table 179 - SMI Referenced Properties/Methods for CIM_EthernetPort

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
DeviceID		Mandatory	
ElementName		Optional	User friendly name
OperationalStatus		Mandatory	
Speed		Mandatory	
MaxSpeed		Mandatory	
PortType		Mandatory	Supported port mode 10BaseT,10-100BaseT, 100BaseT, 1000BaseT, etc.
PortNumber		Mandatory	System level port or bus identification number
NetworkAddresses		Mandatory	MAC addresses
LinkTechnology		Mandatory	Ethernet

16.8.25 CIM_EthernetPortStatistics

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

Table 180 describes class CIM_EthernetPortStatistics.

Table 180 - SMI Referenced Properties/Methods for CIM_EthernetPortStatistics

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Mandatory	
StatisticTime		Optional	

Table 180 - SMI Referenced Properties/Methods for CIM_EthernetPortStatistics

Properties	Flags	Requirement	Description & Notes
BytesTransmitted		Mandatory	
BytesReceived		Mandatory	
PacketsTransmitted		Mandatory	
PacketsReceived		Mandatory	
SymbolErrors		Mandatory	
AlignmentErrors		Optional	
FCSErrors		Optional	
SingleCollisionFrames		Optional	
MultipleCollisionFrames		Optional	
DeferredTransmissions		Optional	
LateCollisions		Optional	
ExcessiveCollisions		Optional	
InternalMACTransmitErrors		Optional	
InternalMACReceiveErrors		Optional	
CarrierSenseErrors		Mandatory	
FrameTooLongs		Optional	
ResetSelectedStats()		Optional	

16.8.26 CIM_FCIPSettings

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

Table 181 describes class CIM_FCIPSettings.

Table 181 - SMI Referenced Properties/Methods for CIM_FCIPSettings

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Optional	User friendly name. In addition, it can be used as a index property for a search or query.
ConnectionUsageFlags		Mandatory	
SpecialFrameTimeout		Mandatory	
KeepAliveTimeout		Mandatory	

16.8.27 CIM_FCPort

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

Table 182 describes class CIM_FCPort.

Table 182 - SMI Referenced Properties/Methods for CIM_FCPort

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
DeviceID		Mandatory	
ElementName		Optional	User Friendly Name
OperationalStatus		Mandatory	
Speed		Mandatory	Speed of zero represents a link not established 1Gb is 1062500000 bps 2Gb is 2125000000 bps 4Gb is 4250000000 bps) 10Gb single channel variants are 10518750000 bps 10Gb four channel variants are 12750000000 bps This is the raw bit rate.
MaxSpeed		Mandatory	The max speed of the Port in Bits per Second using the same algorithm as Speed.

Table 182 - SMI Referenced Properties/Methods for CIM_FCPort

Properties	Flags	Requirement	Description & Notes
PortType		Mandatory	FC-GS Port.Type The specific mode currently enabled for the Port. The values: "N" = Node Port, "NL" = Node Port supporting FC arbitrated loop, "E" = Expansion Port connecting fabric elements (for example, FC switches), "F" = Fabric (element) Port, "FL" = Fabric (element) Port supporting FC arbitrated loop, and "B" = Bridge Port. PortTypes are defined in the ANSI INCITS FC-GS standards. When set to 1 ("Other"), the related property OtherPortType contains a string description of the port's type. PortType is defined to force consistent naming of the 'type' property in subclasses and to guarantee unique enum values for all instances of NetworkPort. A range of values, DMTF_Reserved, has been defined that allows subclasses to override and define their specific port types. Vendor Reserved = 16000..65535 can be used if the PortType is not one already defined in the above enumerations and a vendor subclass is defined specifying the appropriate 'value' and 'valuemap'.
PortNumber		Mandatory	System level port or bus identification number
PermanentAddress		Mandatory	For FibreChannel, it is the Fibre Channel Port WWN.
LinkTechnology		Mandatory	"FC"
SupportedCOS		Mandatory	
ActiveCOS		Optional	
SupportedMaximumTransmissionUnit		Mandatory	
ActiveMaximumTransmissionUnit		Optional	

16.8.28 CIM_FCPortStatistics

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

Table 183 describes class CIM_FCPortStatistics.

Table 183 - SMI Referenced Properties/Methods for CIM_FCPortStatistics

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Optional	

Table 183 - SMI Referenced Properties/Methods for CIM_FCPortStatistics

Properties	Flags	Requirement	Description & Notes
StatisticTime		Mandatory	
BytesTransmitted		Mandatory	
BytesReceived		Mandatory	
PacketsTransmitted		Mandatory	
PacketsReceived		Mandatory	
CRCErrors		Mandatory	
LinkFailures		Mandatory	
PrimitiveSeqProtocol ErrCount		Mandatory	
LossOfSignalCounter		Mandatory	
InvalidTransmission Words		Mandatory	
SampleInterval		Mandatory	
LIPCount		Mandatory	
NOSCount		Mandatory	
ErrorFrames		Mandatory	
DumpedFrames		Mandatory	
LossOfSyncCounter		Mandatory	
FramesTooShort		Mandatory	
FramesTooLong		Mandatory	
AddressErrors		Mandatory	
BufferCreditNotProvi ded		Mandatory	
DelimiterErrors		Mandatory	
EncodingDisparityErr ors		Mandatory	
LinkResetsReceived		Mandatory	
LinkResetsTransmitt ed		Mandatory	
MulticastFramesRec eived		Mandatory	
MulticastFramesTran smitted		Mandatory	
RXBroadcastFrames		Mandatory	

Table 183 - SMI Referenced Properties/Methods for CIM_FCPortStatistics

Properties	Flags	Requirement	Description & Notes
TXBroadcastFrames		Mandatory	
FBSYFrames		Mandatory	
PBSYFrames		Mandatory	
FRJTFrames		Mandatory	
PRJTFrames		Mandatory	
RXClass1Frames		Mandatory	
TXClass1Frames		Mandatory	
RXClass2Frames		Mandatory	
TXClass2Frames		Mandatory	
Class2FBSY		Mandatory	
Class2PBSY		Mandatory	
Class2FRJT		Mandatory	
Class2PRJT		Mandatory	
RXClass3Frames		Mandatory	
TXClass3Frames		Mandatory	
Class3FramesDiscarded		Mandatory	
ResetSelectedStats()		Optional	

16.8.29 CIM_HostedAccessPoint (ComputerSYstem to IPProtocolEndpoint)

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

Table 184 describes class CIM_HostedAccessPoint (ComputerSYstem to IPProtocolEndpoint).

Table 184 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (ComputerSYstem to IPProtocolEndpoint)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

16.8.30 CIM_HostedAccessPoint (ComputerSYstem to TCPProtocolEndpoint)

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

Table 185 describes class CIM_HostedAccessPoint (ComputerSYstem to TCPProtocolEndpoint).

Table 185 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (ComputerSYstem to TCPProtocolEndpoint)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

16.8.31 CIM_HostedAccessPoint (ComputerSystem to ProtocolEndpoint)

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

Table 186 describes class CIM_HostedAccessPoint (ComputerSystem to ProtocolEndpoint).

Table 186 - SMI Referenced Properties/Methods for CIM_HostedAccessPoint (ComputerSystem to ProtocolEndpoint)

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

16.8.32 CIM_HostedNetworkPipe

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

Table 187 describes class CIM_HostedNetworkPipe.

Table 187 - SMI Referenced Properties/Methods for CIM_HostedNetworkPipe

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

16.8.33 CIM_IPEndpointStatistics

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

Table 188 describes class CIM_IPEndpointStatistics.

Table 188 - SMI Referenced Properties/Methods for CIM_IPEndpointStatistics

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Optional	User friendly name. In addition, it can be used as a index property for a search or query.
StatisticTime		Mandatory	
ReceivedPDUs		Mandatory	
ReceivedPDUHeaderErrors		Mandatory	
ReceivedPDUAddressErrors		Mandatory	
ReceivedPDUForwards		Mandatory	
ReceivedPDUUnknownProtocolErrors		Mandatory	
ReceivedPDUDiscards		Mandatory	
PDUDelivers		Mandatory	
SentPDUs		Mandatory	
SentPDUDiscards		Mandatory	
SentPDUNoRouteErrors		Mandatory	
ReassemblyRequired		Mandatory	

Table 188 - SMI Referenced Properties/Methods for CIM_IPEndpointStatistics

Properties	Flags	Requirement	Description & Notes
ReassembledPackets		Mandatory	
ReassemblyFailed		Mandatory	
Fragmentation		Mandatory	
FragmentationFails		Mandatory	
FragmentedPDUsCreates		Mandatory	
RouteEntriesDiscards		Mandatory	
ResetSelectedStats()		Optional	

16.8.34 CIM_IPProtocolEndpoint

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

Table 189 describes class CIM_IPProtocolEndpoint.

Table 189 - SMI Referenced Properties/Methods for CIM_IPProtocolEndpoint

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	
NameFormat		Mandatory	
IPv4Address		Mandatory	
IPv6Address		Mandatory	
SubnetMask		Mandatory	
PrefixLength		Optional	
ProtocolIFType		Mandatory	IPv4, IPv6, IPv4/v6

16.8.35 CIM_IPSettings

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

Table 190 describes class CIM_IPSettings.

Table 190 - SMI Referenced Properties/Methods for CIM_IPSettings

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Optional	User friendly name. In addition, it can be used as a index property for a search or query.
FragmentationTimeout		Mandatory	
EnableIPForwarding		Mandatory	

16.8.36 CIM_LANEndpoint

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

Table 191 describes class CIM_LANEndpoint.

Table 191 - SMI Referenced Properties/Methods for CIM_LANEndpoint

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	
NameFormat		Mandatory	
ProtocolIFType		Mandatory	Fibrechannel, Fcip

16.8.37 CIM_Network

Created By: Static
Modified By: Static
Deleted By: Static

Requirement: Mandatory

Table 192 describes class CIM_Network.

Table 192 - SMI Referenced Properties/Methods for CIM_Network

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Name		Mandatory	IP Address
NameFormat		Mandatory	IP Address

16.8.38 CIM_NetworkPipe

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

Table 193 describes class CIM_NetworkPipe.

Table 193 - SMI Referenced Properties/Methods for CIM_NetworkPipe

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
Directionality		Optional	
OperationalStatus		Optional	
AggregationBehavior		Optional	
EnabledState		Optional	
RequestedState		Optional	

16.8.39 CIM_NetworkPipeComposition

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

Table 194 describes class CIM_NetworkPipeComposition.

Table 194 - SMI Referenced Properties/Methods for CIM_NetworkPipeComposition

Properties	Flags	Requirement	Description & Notes
AggregationSequence		Mandatory	
GroupComponent		Mandatory	
PartComponent		Mandatory	

16.8.40 CIM_ProtocolEndpoint

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

Table 195 describes class CIM_ProtocolEndpoint.

Table 195 - SMI Referenced Properties/Methods for CIM_ProtocolEndpoint

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	
NameFormat		Mandatory	
ProtocolIFType		Mandatory	Fibrechannel, Fcip

16.8.41 CIM_RemotePort

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

Table 196 describes class CIM_RemotePort.

Table 196 - SMI Referenced Properties/Methods for CIM_RemotePort

Properties	Flags	Requirement	Description & Notes
SystemName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	Opaque
AccessInfo		Mandatory	
InfoFormat		Mandatory	
OtherInfoFormatDescription		Mandatory	WWN
PortInfo		Optional	WWN or TCP port number
PortProtocol		Mandatory	
OtherProtocolDescription		Mandatory	

16.8.42 CIM_RemoteServiceAccessPoint

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

Table 197 describes class CIM_RemoteServiceAccessPoint.

Table 197 - SMI Referenced Properties/Methods for CIM_RemoteServiceAccessPoint

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	
AccessInfo		Mandatory	
InfoFormat		Mandatory	IPv4 Address OR IPv6 Address
OtherInfoFormatDescription		Optional	

16.8.43 CIM_SystemDevice (System to EthernetPort)

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

Table 198 describes class CIM_SystemDevice (System to EthernetPort).

Table 198 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to EthernetPort)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

16.8.44 CIM_SystemDevice (System to FCPort)

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

Table 199 describes class CIM_SystemDevice (System to FCPort).

Table 199 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to FCPort)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

16.8.45 CIM_TCPEndpointStatistics

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

Table 200 describes class CIM_TCPEndpointStatistics.

Table 200 - SMI Referenced Properties/Methods for CIM_TCPEndpointStatistics

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
ElementName		Optional	User friendly name. In addition, it can be used as a index property for a search or query.
StatisticTime		Mandatory	
ReceivedSegmentsInError		Mandatory	
SentResetSegments		Mandatory	
ResetSelectedStats()		Optional	

16.8.46 CIM_TCPProtocolEndpoint

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

Table 201 describes class CIM_TCPProtocolEndpoint.

Table 201 - SMI Referenced Properties/Methods for CIM_TCPProtocolEndpoint

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
Name		Mandatory	
NameFormat		Optional	
PortNumber		Optional	
ProtocolIFType		Mandatory	

16.8.47 CIM_TCPSettings

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

Table 202 describes class CIM_TCPSettings.

Table 202 - SMI Referenced Properties/Methods for CIM_TCPSettings

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Optional	User friendly name. In addition, it can be used as a index property for a search or query.
RetransmissionTimeoutAlgorithm		Mandatory	
RetransmissionTimeoutMin		Mandatory	
RetransmissionTimeoutMax		Mandatory	

16.8.48 CIM_TCPStatisticalData

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

Table 203 describes class CIM_TCPStatisticalData.

Table 203 - SMI Referenced Properties/Methods for CIM_TCPStatisticalData

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Opaque
ElementName		Optional	User Friendly Name
StatisticTime		Mandatory	
ActiveOpenConnections		Mandatory	
PassiveOpenConnections		Mandatory	
AttemptsFails		Mandatory	
EstablishedResets		Mandatory	
EstablishedConnections		Mandatory	
ReceivedSegments		Mandatory	
SentSegments		Mandatory	

Table 203 - SMI Referenced Properties/Methods for CIM_TCPStatisticalData

Properties	Flags	Requirement	Description & Notes
RetransmittedSegments		Mandatory	
ReceivedSegmentsInError		Mandatory	
SentResetSegments		Mandatory	
ResetSelectedStats()		Mandatory	

EXPERIMENTAL

DEPRECATED

Clause 17: Router Profile

The functionality of the Router Profile (Section 7.3.4.3 of SMI-S 1.0.2) modeled FC/SCSI routers used in some tape libraries.

DEPRECATED

EXPERIMENTAL

Clause 18: iSCSI Gateway Profile

18.1 Description

This profile exposes FC LUNs through an iSCSI Node. It includes the discovery of the exposed FC LUNs to the iSCSI LUN (mapping) and the configuration of the mapping. The mapping uses the methods ExposePaths and HidePaths in StorageConfigurationService, but instead of exposing the LUN to a Host, it is exposing a LUN from a Fibre Channel Fabric to an iSCSI Fabric.

It supports the profile of the:

1) iSCSI Target Ports Subprofile which presents discovery and configuration of the iSCSI nodes and iSCSI Ports (Target Portal Groups) and the discovery and configuration of exposing storage volumes through iSCSI Nodes.

2) Fibre Channel Initiator Port Subprofile which presents the FC Initiator and attached Volume.

3) Host Discovered Resources Profile which optionally presents the storage accessible through a particular FC Initiator.

4) DMTF Ethernet Port Profile which exposes the discovery of an Ethernet Port.

5) DMTF IP Interface Profile which exposes the discovery and configuration of an IP Interface.

6) FC Target Volume is identified as the concatenation of the FC Target Port and LUN.

In Figure 25, the instance diagram for the iSCSI to FC Gateway is shown.

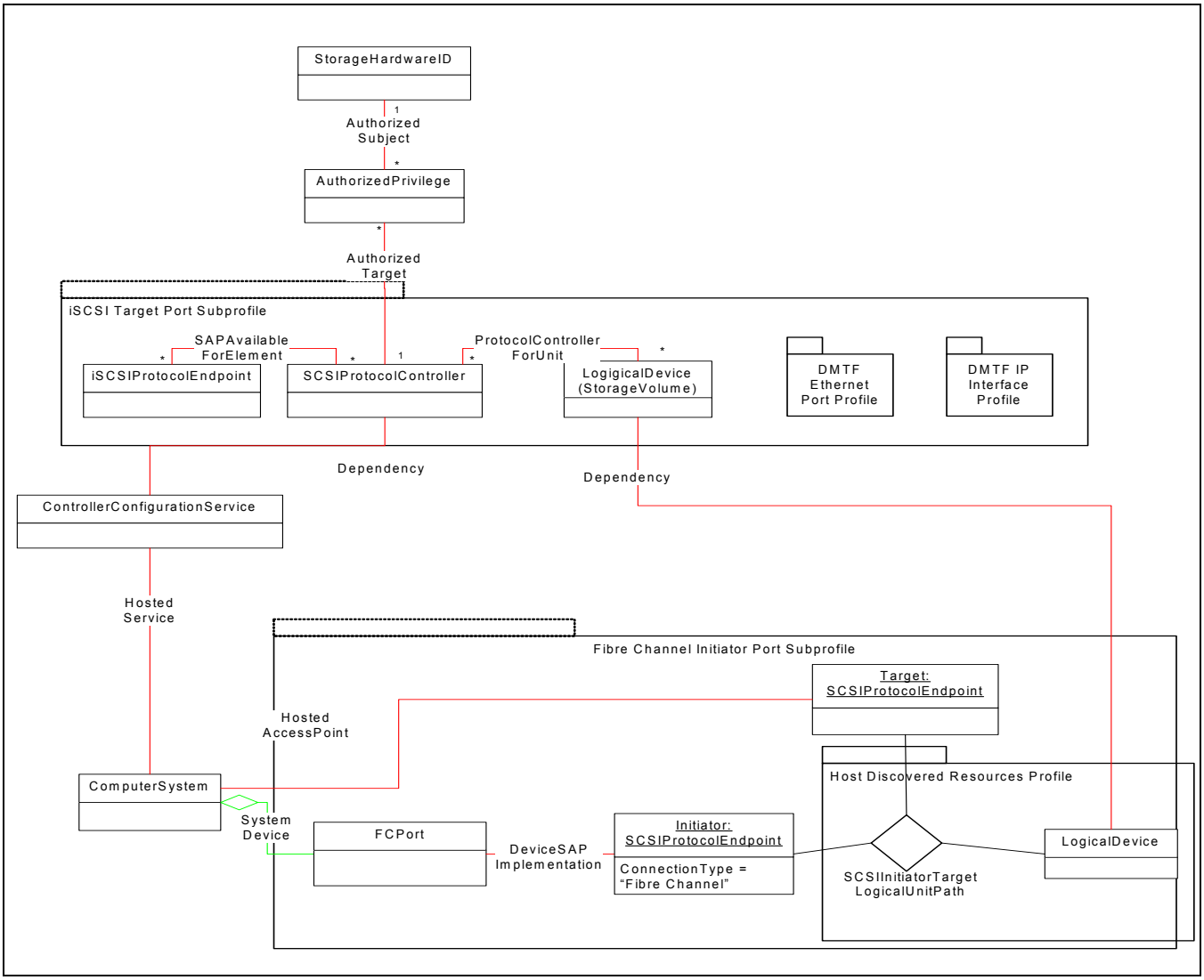

Figure 25 - iSCSI to FC Gateway Instance

18.2 Health and Fault Management Consideration

None

18.3 Cascading Considerations

None

18.4 Supported Profiles, Subprofiles, and Packages

Profile Name: iSCSI to FC Gateway

Version: 1.2.0

Organization: SNIA

CIM Schema Version: TBD

Table 205 describes the related profiles for iSCSI to FC Gateway.

Table 204 - Related Profiles for iSCSI to FC Gateway

Profile Name	Organization	Version	Requirement	Description
Access Points	SNIA	1.3.0	Optional	
iSCSI Target Ports	SNIA	1.2.0	Mandatory	
FC Initiator Ports	SNIA	1.3.0	Mandatory	
Ethernet Port	DMTF	1.0	Mandatory	
IP Interface	DMTF	1.0.0	Mandatory	
Host Discovered Resources	SNIA	1.2.0	Optional	

18.5 Methods of the Profile

The mapping uses the methods ExposePaths and HidePaths in ControllerConfigurationService to expose or hide the LUN from a Fibre Channel Fabric to an iSCSI Fabric.

18.5.0.0.1 HidePaths

```
uint32 HidePaths (
OUT CIM_ConcreteJob REF Job
IN string LUNames[]
IN string InitiatorPortIDs[]
IN string TargetPortIDs[]
IN/OUT CIM_SCSIProtocolController REF ProtocolControllers[] )
```

18.5.0.0.2 ExposePaths

```
Uint32 ExposePaths (
OUT CIM_ConcreteJob REF Job,
IN string LUNames[]
IN string InitiatorPortIDs[]
IN string TargetPortIDs[]
IN string DeviceNumber[]
IN uint16 DeviceAccess[]
IN/OUT CIM_SCSIProtocolController REF ProtocolControllers[])
```

18.6 Client Considerations and Recipes

None

18.7 CIM Elements

Table 205 describes the CIM elements for iSCSI to FC Gateway.

Table 205 - CIM Elements for iSCSI to FC Gateway

Element Name	Requirement	Description
18.7.1 CIM_AuthorizedPrivilege	Mandatory	
18.7.2 CIM_AuthorizedSubject	Mandatory	
18.7.3 CIM_AuthorizedTarget	Mandatory	
18.7.4 CIM_ControllerConfigurationService	Mandatory	
18.7.5 CIM_ControllerConfigurationService	Mandatory	
18.7.6 CIM_Dependency (Associates ControllerConfiguirationService and ProtocolController)	Mandatory	
18.7.7 CIM_Dependency (Associates PrivilegeManagementService and AuthorizedPrivilege)	Mandatory	
18.7.8 CIM_HostedService (Associates ComputerSystem and ControllerConfigurationService)	Mandatory	
18.7.9 CIM_HostedService (Associates ComputerSystem and PrivilegeManagementService)	Mandatory	
18.7.10 CIM_LogicalDevice	Mandatory	
18.7.11 CIM_PrivilegeManagementService	Mandatory	
18.7.12 CIM_ProtocolControllerForUnit	Mandatory	
18.7.13 CIM_SAPAvailableForElement	Mandatory	
18.7.14 CIM_SCSIProtocolController	Mandatory	
18.7.15 CIM_SCSIProtocolEndpoint	Mandatory	
18.7.16 CIM_StorageHardwareID	Mandatory	
18.7.17 CIM_StorageHardwareIDManagementService	Mandatory	

18.7.1 CIM_AuthorizedPrivilege

Created By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths

Modified By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths

Deleted By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths

Requirement: Mandatory

Table 206 describes class CIM_AuthorizedPrivilege.

Table 206 - SMI Referenced Properties/Methods for CIM_AuthorizedPrivilege

Properties	Requirement	Description & Notes
InstanceID	Mandatory	Opaque and unique identifer
ElementName	Optional	User friendly name
PrivilegeGranted	Mandatory	Indicates if the privilege is granted or not
Activities	Mandatory	For SMI-S, must be "Read", "Write"

18.7.2 CIM_AuthorizedSubject

Created By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths

Modified By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths

Deleted By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths

Requirement: Mandatory

Table 207 describes class CIM_AuthorizedSubject.

Table 207 - SMI Referenced Properties/Methods for CIM_AuthorizedSubject

Properties	Requirement	Description & Notes
PrivilegedElement	Mandatory	The Subject for which Privileges are granted or denied
Privilege	Mandatory	The Privilege either granted or denied to an Identity or group of Identities collected by a Role.

18.7.3 CIM_AuthorizedTarget

Created By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths

Modified By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths

Deleted By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths

Requirement: Mandatory

Table 208 describes class CIM_AuthorizedTarget.

Table 208 - SMI Referenced Properties/Methods for CIM_AuthorizedTarget

Properties	Requirement	Description & Notes
TargetElement	Mandatory	The target set of resources to which the Privilege applies
Privilege	Mandatory	The Privilege affecting the target resource

18.7.4 CIM_ControllerConfigurationService

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

Table 209 describes class CIM_ControllerConfigurationService.

Table 209 - SMI Referenced Properties/Methods for CIM_ControllerConfigurationService

Properties	Requirement	Description & Notes
SystemCreationClassName	Mandatory	The scoping System CreationClassName
SystemName	Mandatory	The scoping System Name
CreationClassName	Mandatory	The name of the concrete subclass
Name	Mandatory	Unique identifer for the Service
ExposePaths()	Mandatory	
HidePaths()	Mandatory	
ExposeDefaultLUs()	Mandatory	
HideDefaultLUs()	Mandatory	
DeleteProtocolController()	Mandatory	

18.7.5 CIM_ControllerConfigurationService

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

Table 210 describes class CIM_ControllerConfigurationService.

Table 210 - SMI Referenced Properties/Methods for CIM_ControllerConfigurationService

Properties	Requirement	Description & Notes
SystemCreationClassName	Mandatory	The scoping System CreationClassName
SystemName	Mandatory	The scoping System Name
CreationClassName	Mandatory	The name of the concrete subclass
Name	Mandatory	Uniquely identifies the Service

18.7.6 CIM_Dependency (Associates ControllerConfiguirationService and ProtocolController)

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

Table 211 describes class CIM_Dependency (Associates ControllerConfiguirationService and ProtocolController).

Table 211 - SMI Referenced Properties/Methods for CIM_Dependency (Associates ControllerConfiguirationService and ProtocolController)

Properties	Requirement	Description & Notes
Dependent	Mandatory	
Antecedent	Mandatory	

18.7.7 CIM_Dependency (Associates PrivilegeManagementService and AuthorizedPrivilege)

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

Table 212 describes class CIM_Dependency (Associates PrivilegeManagementService and AuthorizedPrivilege).

Table 212 - SMI Referenced Properties/Methods for CIM_Dependency (Associates PrivilegeManagementService and AuthorizedPrivilege)

Properties	Requirement	Description & Notes
Antecedent	Mandatory	
Dependent	Mandatory	

18.7.8 CIM_HostedService (Associates ComputerSystem and ControllerConfigurationService)

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

Table 213 describes class CIM_HostedService (Associates ComputerSystem and ControllerConfigurationService).

Table 213 - SMI Referenced Properties/Methods for CIM_HostedService (Associates ComputerSystem and ControllerConfigurationService)

Properties	Requirement	Description & Notes
Antecedent	Mandatory	
Dependent	Mandatory	

18.7.9 CIM_HostedService (Associates ComputerSystem and PrivilegeManagementService)

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

Table 214 describes class CIM_HostedService (Associates ComputerSystem and PrivilegeManagementService).

Table 214 - SMI Referenced Properties/Methods for CIM_HostedService (Associates ComputerSystem and PrivilegeManagementService)

Properties	Requirement	Description & Notes
Antecedent	Mandatory	
Dependent	Mandatory	

18.7.10 CIM_LogicalDevice

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

Table 215 describes class CIM_LogicalDevice.

Table 215 - SMI Referenced Properties/Methods for CIM_LogicalDevice

Properties	Requirement	Description & Notes
SystemCreationClassName	Mandatory	The scoping System CreationClassName
CreationClassName	Mandatory	The name of the concrete subclass
SystemName	Mandatory	The scoping System Name
DeviceID	Mandatory	Unique identifer

18.7.11 CIM_PrivilegeManagementService

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

Table 216 describes class CIM_PrivilegeManagementService.

Table 216 - SMI Referenced Properties/Methods for CIM_PrivilegeManagementService

Properties	Requirement	Description & Notes
SystemCreationClassName	Mandatory	The scoping System CreationClassName
CreationClassName	Mandatory	The name of the concrete subclass
SystemName	Mandatory	The scoping System Name
Name	Mandatory	Uniquely identifies the Service
ElementName	Mandatory	User friendly name
AssignAccess()	Mandatory	
RemoveAccess()	Mandatory	

18.7.12 CIM_ProtocolControllerForUnit

Created By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths, CIM_ControllerConfigurationService.ExposeDefaultLUs, CIM_ControllerConfigurationService.HideDefaultLUs

Modified By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths, CIM_ControllerConfigurationService.ExposeDefaultLUs, CIM_ControllerConfigurationService.HideDefaultLUs

Deleted By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths, CIM_ControllerConfigurationService.ExposeDefaultLUs, CIM_ControllerConfigurationService.HideDefaultLUs

Requirement: Mandatory

Table 217 describes class CIM_ProtocolControllerForUnit.

Table 217 - SMI Referenced Properties/Methods for CIM_ProtocolControllerForUnit

Properties	Requirement	Description & Notes
DeviceNumber	Mandatory	Address (e.g. LUN) of the associated Device. Shall be formatted as unseparated uppercase hexadecimal digits, with no leading 0x.
DeviceAccess	Mandatory	The access rights granted to the referenced logical unit as exposed through referenced ProtocolController
Antecedent	Mandatory	
Dependent	Mandatory	

18.7.13 CIM_SAPAvailableForElement

Created By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths, CIM_ControllerConfigurationService.ExposeDefaultLUs, CIM_ControllerConfigurationService.HideDefaultLUs

Modified By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths, CIM_ControllerConfigurationService.ExposeDefaultLUs, CIM_ControllerConfigurationService.HideDefaultLUs

Deleted By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths, CIM_ControllerConfigurationService.ExposeDefaultLUs, CIM_ControllerConfigurationService.HideDefaultLUs

Requirement: Mandatory

Table 218 describes class CIM_SAPAvailableForElement.

Table 218 - SMI Referenced Properties/Methods for CIM_SAPAvailableForElement

Properties	Requirement	Description & Notes
AvailableSAP	Mandatory	
ManagedElement	Mandatory	

18.7.14 CIM_SCSIProtocolController

Created By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths

Modified By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths

Deleted By: Extrinsics: CIM_ControllerConfigurationService.ExposePaths, CIM_ControllerConfigurationService.HidePaths

Requirement: Mandatory

Table 219 describes class CIM_SCSIProtocolController.

Table 219 - SMI Referenced Properties/Methods for CIM_SCSIProtocolController

Properties	Requirement	Description & Notes
SystemCreationClassName	Mandatory	The scoping System CreationClassName
CreationClassName	Mandatory	The name of the concrete subclass
SystemName	Mandatory	The scoping System's Name
DeviceID	Mandatory	Unique name for the ProtocolController

18.7.15 CIM_SCSIProtocolEndpoint

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

Table 220 describes class CIM_SCSIProtocolEndpoint.

Table 220 - SMI Referenced Properties/Methods for CIM_SCSIProtocolEndpoint

Properties	Requirement	Description & Notes
SystemCreationClassName	Mandatory	The scoping System CreationClassName
SystemName	Mandatory	The scoping System Name
CreationClassName	Mandatory	The name of the concrete subclass

18.7.16 CIM_StorageHardwareID

Created By: Extrinsic: CIM_StorageHardwareIDManagementService.CreateStorageHardwareID
Modified By: Static
Deleted By: Extrinsic: CIM_StorageHardwareIDManagementService.DeleteStorageHardwareID
Requirement: Mandatory

Table 221 describes class CIM_StorageHardwareID.

Table 221 - SMI Referenced Properties/Methods for CIM_StorageHardwareID

Properties	Requirement	Description & Notes
InstanceID	Mandatory	Opaque and unique identifier
StorageID	Mandatory	The worldwide unique ID
IDType	Mandatory	StorageID type. Values are Other, PortWWN, NodeWWN, Hostname, and iSCSI Name

18.7.17 CIM_StorageHardwareIDManagementService

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

Table 222 describes class CIM_StorageHardwareIDManagementService.

Table 222 - SMI Referenced Properties/Methods for CIM_StorageHardwareIDManagementService

Properties	Requirement	Description & Notes
SystemCreationClassName	Mandatory	The scoping System CreationClassName
SystemName	Mandatory	The scoping System Name
CreationClassName	Mandatory	The name of the concrete subclass
Name	Mandatory	Uniquely identifies the Service
CreateStorageHardwareID()	Mandatory	
DeleteStorageHardwareID()	Mandatory	
CreateHardwareIDCollection()	Optional	
AddHardwareIDsToCollection()	Optional	

EXPERIMENTAL