Storage Management Technical Specification, Part 7 Media Libraries

Version 1.4.0, Revision 6

Abstract: This SNIA Technical Position defines an interface between WBEM-capable clients and servers for the secure, extensible, and interoperable management of networked storage.

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

SNIA Technical Position

17 May 2010

Revision History

Revision 1

Date

5 December 2007

SCRs Incorporated and other changes

None.

Comments

Editorial notes and DRAFT material are displayed.

Revision 2

Date

29 February 2008

SCRs Incorporated and other changes

Clause 11: Virtual Tape Library System Profile

- Added support for Element Counting as a subprofile of VTL (SML-SMIS-SCR0015)

- Replaced references to LUNs with StorageVolume (SML-SMIS-SCR00022)

Clause 13: Partitioned Tape Library Profile (no SCR, just a ballot in SML)

- New Profile added to SMI-S 1.4.0

Comments

Editorial notes and DRAFT material are displayed.

Revision 3

Date

23 May 2008

SCRs Incorporated and other changes

Partitioned Tape Library Profile (SMIS-140-Draft-SCR00013.000)

- Updated first figure Partitioned Tape Library (PTL) System model by deleting the PartitionUnit and PartitionUnitCapabilities from the model.

- Sections 11.1.3 PTL Configuration and Section 11.1.4 PTL Configuration Methods are added that describe the PTL model and configuration service methods.

- Section 11.4 Profiles, Subprofiles, and Packages table was added.

- Section 11.7 CIM Elements tables were added.

- **PROMOTED**: Promoted the profile from Draft to Experimental.

Comments

Editorial notes are displayed.

Revision 4

Date

23 October 2008

SCRs Incorporated and other changes

PartitionTapeLibrary

- Removed Physical Package from supported Profiles per 8/18/2008 ballot

VTLCopy

- Removed indications from profile per Alex

VTL

- Migrated 1.3 VTL to 1.4

Comments

Editorial notes and DRAFT material are not displayed.

Revision 5

Date

9 December 2009

SCRs Incorporated and other changes

None

Comments

The errata SCRs listed above indicate changes made since this part of SMI-S 1.4 Revision 4 was reviewed and approved by SNIA membership and subsequently published as a SNIA Technical Position.

SMIS-140-Errata-SCR00001 through SMIS-140-Errata-SCR00060 (approved SCRs) have been incorporated into SMI-S 1.4 Revision 5.

Revision 6

Date

17 May 2010

SCRs Incorporated and other changes

None

Comments

The errata SCRs listed above indicate changes made since this part of SMI-S 1.4 Revision 4 was reviewed and approved by SNIA membership and subsequently published as a SNIA Technical Position.

SMIS-140-Errata-SCR00061 through SMIS-140-Errata-SCR00067 (approved SCRs) have been incorporated into SMI-S 1.4 Revision 6.

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 Storage Networking Industry Association (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 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.

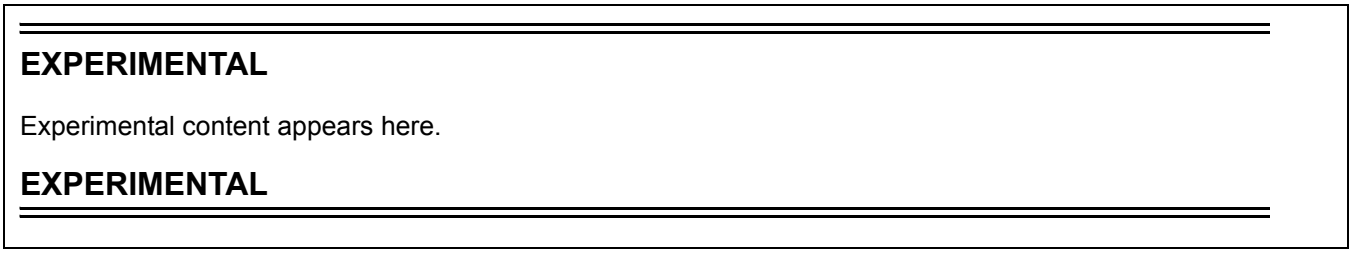

Figure 1 - Experimental Maturity Level Tag

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.

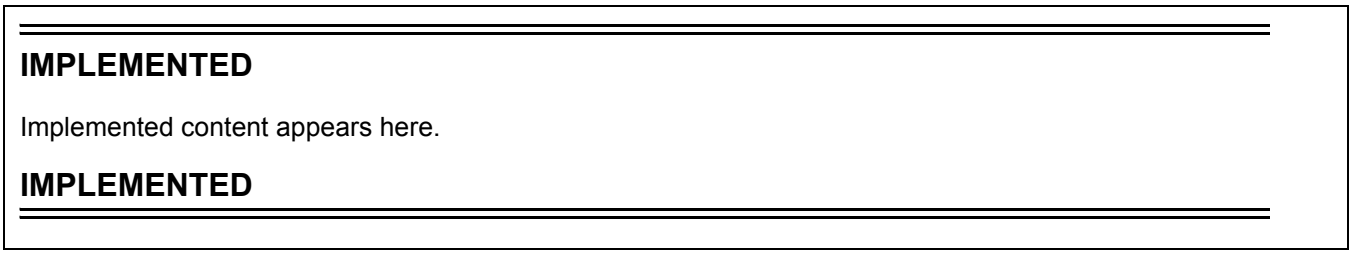

Figure 2 - Implemented Maturity Level Tag

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 Storage Library Profile and related subprofiles defined in this part provide a standard CIM interface to monitor and control various aspects of removable media libraries including tape libraries. Once a library supports this specification, any SMI-S client based on this specification can discover a tape library, determine its capacity, perform inventory, monitor status, move tapes and perform other configuration and control operations. This specification also standardizes library specific life-cycle and alert indications that are delivered to a client asynchronously, once a client subscribes to these indications.

This part is Part 7 (Media Libraries) of the SMI-S specification listed below. While *Storage Management Technical Specification, Part 1 Common Architecture, 1.4.0 Rev 6* describes SMI-S concepts and terms, some of the profiles and subprofiles referenced in the Storage Library profile are specified in *Storage Management Technical Specification, Part 2 Common Profiles, 1.4.0 Rev 6*.

Parts of this Standard

This standard is subdivided in the following parts:

- *Storage Management Technical Specification, Overview, 1.4.0 Rev 6*
- *Storage Management Technical Specification, Part 1 Common Architecture, 1.4.0 Rev 6*
- *Storage Management Technical Specification, Part 2 Common Profiles, 1.4.0 Rev 6*
- *Storage Management Technical Specification, Part 3 Block Devices, 1.4.0 Rev 6*
- *Storage Management Technical Specification, Part 4 File Systems, 1.4.0 Rev 6*
- *Storage Management Technical Specification, Part 5 Fabric, 1.4.0 Rev 6*
- *Storage Management Technical Specification, Part 6 Host Elements, 1.4.0 Rev 6*
- *Storage Management Technical Specification, Part 7 Media Libraries, 1.4.0 Rev 6*

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
EMC	Mike Hadavi
	Mike Thompson
Hewlett Packard	Steve Peters
Hitachi Data Systems	Eric Hibbard
	Steve Quinn
IBM	Krishna Harathi
	Mike Walker
Olocity	Scott Baker
Pillar	Gary Steffens
Symantec	Steve Hand
	Paul von Behren

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, 425 Market Street, Suite 1020, San Francisco, CA 94105, U.S.A.

Clause 1: Scope

This version of the Storage Library Profile models various details of the following objects of the media library for monitoring.

- Library
- Drives
- Changer Devices
- Slots
- IO Slots
- SCSI Interfaces and SCSI and FC Target Ports
- Physical Tapes
- Physical Package
- Magazines

In general, a CIM client can monitor the health and status of the above objects as well as get alert, status change and lifecycle CIM indications. In addition, a client can control the movement of media in a library using this specification.

The future versions of this specification shall address partitioned tape libraries and virtual tape libraries. Note that the experimental subprofile modelling partitioned tape libraries and virtual tape libraries in the previous version of this specification has been withdrawn and hence is now omitted from this specification.

Clause 2: Normative References

2.1 General

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.

2.2 Approved references

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

2.3 References under development

Storage Management Technical Specification, Part 1 Common Architecture, 1.4.0 Rev 6

Storage Management Technical Specification, Part 2 Common Profiles, 1.4.0 Rev 6

Storage Management Technical Specification, Part 3 Block Devices, 1.4.0 Rev 6

ISO/IEC 14776-452, SCSI Primary Commands - 3 (SPC-3) [ANSI INCITS.351-2005]

2.4 Other references

DMTF DSP0214:2004 CIM Operations over HTTP

Clause 3: Terms and definitions

3.1 General

For the purposes of this document, the terms and definitions given in *Storage Management Technical Specification, Part 1 Common Architecture, 1.4.0 Rev 6* and the following apply.

3.2 Definitions

3.2.1 Changer Device

The robotic arm and control logic within a storage media library that moves media from one location to another.

3.2.2 Media Access Device

A device that performs read and write operations on media. In tape libraries, it is the tape drive.

3.2.3 Storage Media Location

Various locations within a media library where the physical media can be placed. These include the changer devices, the media access devices, physical slots or magazines, and I/O slots.

3.2.4 Storage Media Library

A library in which a large number of removable media can be stored and retrieved. A library also contains a limited number of media access devices for reading and writing to the media. A changer device within the library moves the media between a stored location and drive or between two locations. The drives, changers and the library are controlled by a host typically via the SCSI and/or FC ports, but other types of ports are possible. A storage media library typically is a tape library.

3.2.5 Limited Access Port

An operator-accessible window of a storage media library through which physical media is fed into the library or physical media can be retrieved out of a library. A Limited access port is also known as an I/O Port, Import Export Port, Mailslot, etc.

3.2.6 Library Capacity

The capacity of a storage media library is measured in terms of the number of physical media it can hold.

3.2.7 Magazine

A magazine is a container that holds multiple physical media. Some storage media libraries have magazines that fit into the physical slot instead of single media.

STABLE

Clause 4: Storage Library Profile

4.1 Description

The schema for a storage library provides the classes and associations necessary to represent various forms of removable media libraries. This profile is based upon the CIM 2.12.1 model and defines the subset of classes that supply the necessary information for robotic storage libraries.

This profile further describes how the classes are to be used to satisfy various use cases and offers suggestions to agent implementers and client application developers. Detailed descriptions of classes are from the CIM 2.12.1 schema.

The relevant objects for a storage library should be instantiated in the name space of the provider (or agent) for a storage library resource. Whenever an instance of a class for a resource may exist in multiple name spaces a *durable name* is defined to aid clients in correlating the objects across name spaces. For storage libraries, durable names are defined for the following resources:

- ChangerDevice
- ComputerSystem
- MediaAccessDevice

The durable names are defined in a following subsection of this profile. All other objects do not require durable names and have instances within a single name space.

4.1.1 Instance Diagrams

The following instance diagrams represent five related views of the storage library profile:

a) System Level

b) MediaAccessDevice and its physical and logical relationships

c) ChangerDevice and its connections to SoftwareIdentity, ProtocolController, and StorageMediaLocation

d) StorageMediaLocation and its relationship to PhysicalMedia and other physical classes

e) StorageMediaLocation and its required Realizes relationships.

4.1.2 System Level View

Figure 5 shows the required components for a ComputerSystem. Note that LogicalDevice subclasses shall be associated with ComputerSystem via SystemDevice.

Note: Classes using a red outline and associations using a dotted outline represent optional components that have been included in the diagram as an aid to understanding.

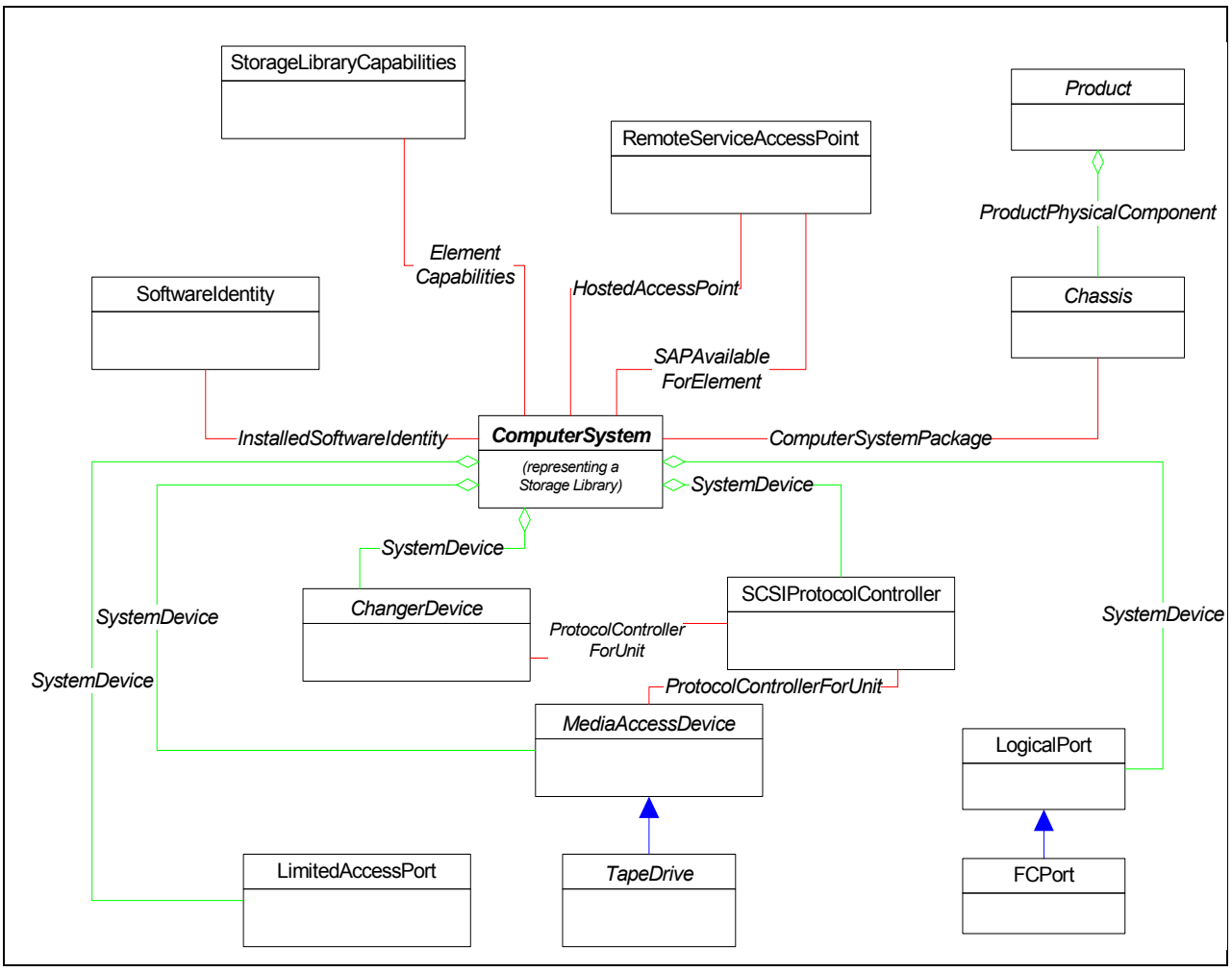

Figure 5 - Storage Library-centric Instance Diagram

4.1.3 MediaAccessDevice-centric View

Figure 6 shows the required classes related to MediaAccessDevice. Though not shown in this figure, both MediaAccessDevice and ProtocolController are connected to a ComputerSystem instance through the SystemDevice association. In some libraries, notably small autoloaders, external hosts access a library's ChangerDevice through the ProtocolController of a MediaAccessDevice. For such libraries, an additional ProtocolControllerForUnit association should be instantiated between the MediaAccessDevice's ProtocolController and the affected ChangerDevice. ProtocolControllerForUnit is a many-to-many association, so a single ProtocolController can be connected to multiple LogicalDevices if this accurately represents a library's configuration.

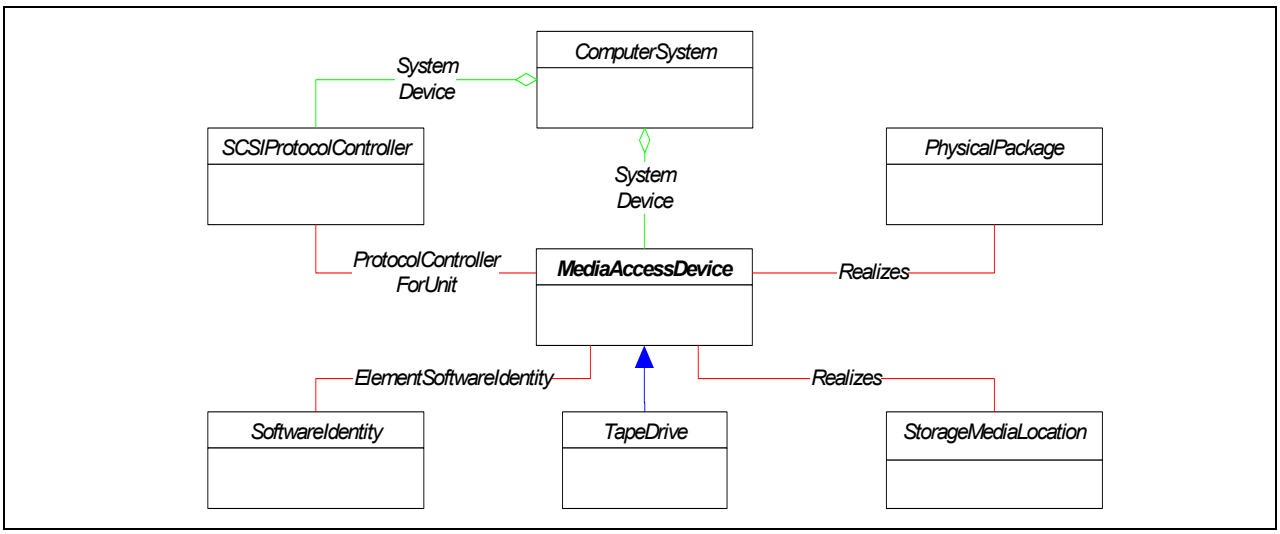

Figure 6 - MediaAccessDevice-centric Instance Diagram

4.1.4 ChangerDevice-centric View

Figure 7 shows the required classes related to ChangerDevice.

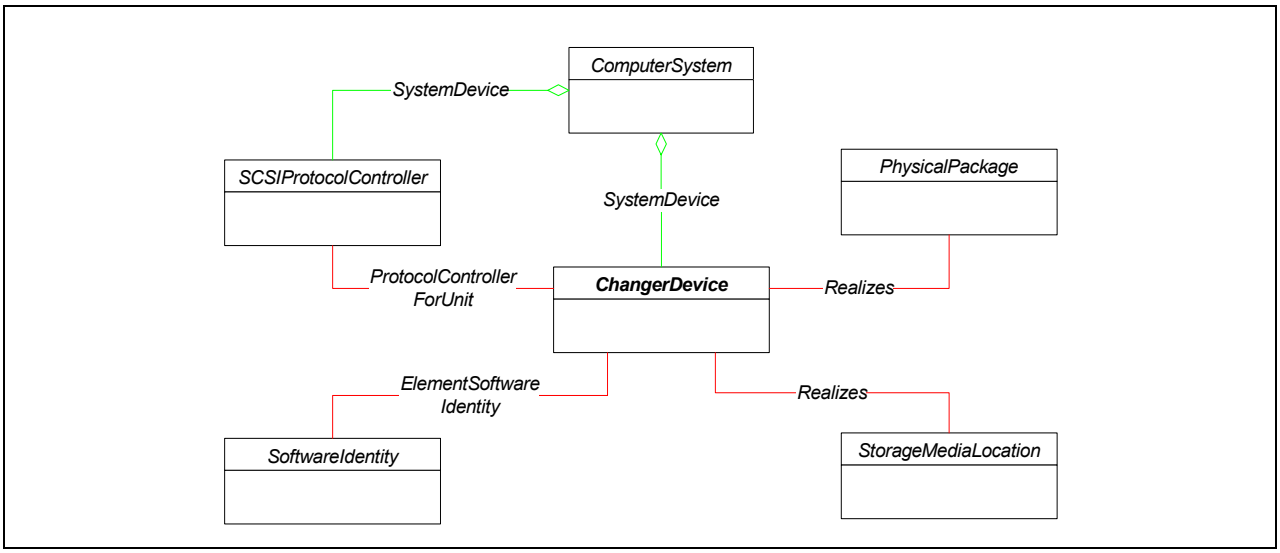

Figure 7 - ChangerDevice-centric Instance Diagram

4.1.5 Physical View

Figure 8 shows important physical components of a storage library and how they relate. With regard to StorageMediaLocation and Magazine, one of two implementation alternatives shall be selected:

a) Instantiate multiple Magazines associated to Chassis via Container, then instantiate StorageMediaLocations that are contained (again via Container) within each Magazine;

b) Instantiate multiple StorageMediaLocations directly associated to Chassis via Container, without the use of Magazines. Other optional classes, such as Panel, can also be used to group StorageMediaLocations, but this is not mandatory.

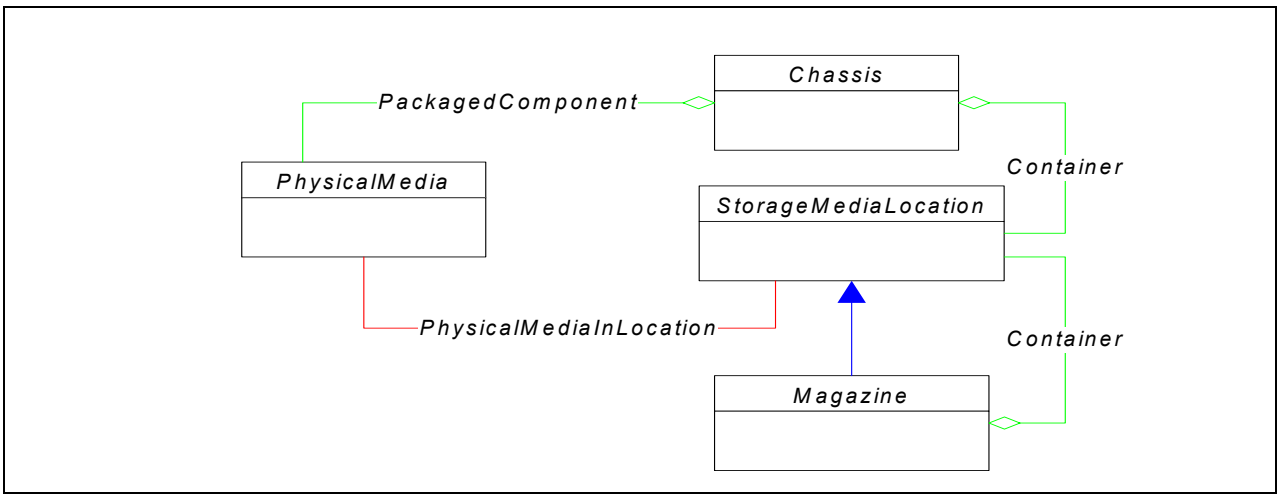

Figure 8 - Physical View Instance Diagram

4.1.6 StorageMediaLocation Instance Diagram

Figure 9 shows relationships between various LogicalDevices (i.e., MediaAccessDevices, LimitedAccessPort, and ChangerDevice) and StorageMediaLocation. For each LogicalDevice that can hold media, at least one StorageMediaLocation shall be associated via Realizes.

The figure also shows how PhysicalMedia is conceptually placed “inside” a LogicalDevice by associating PhysicalMedia with a StorageMediaLocation that Realizes a LogicalDevice (see Figure 9). All tapes, irrespective of the location, are associated with the chassis using PackagedComponent.

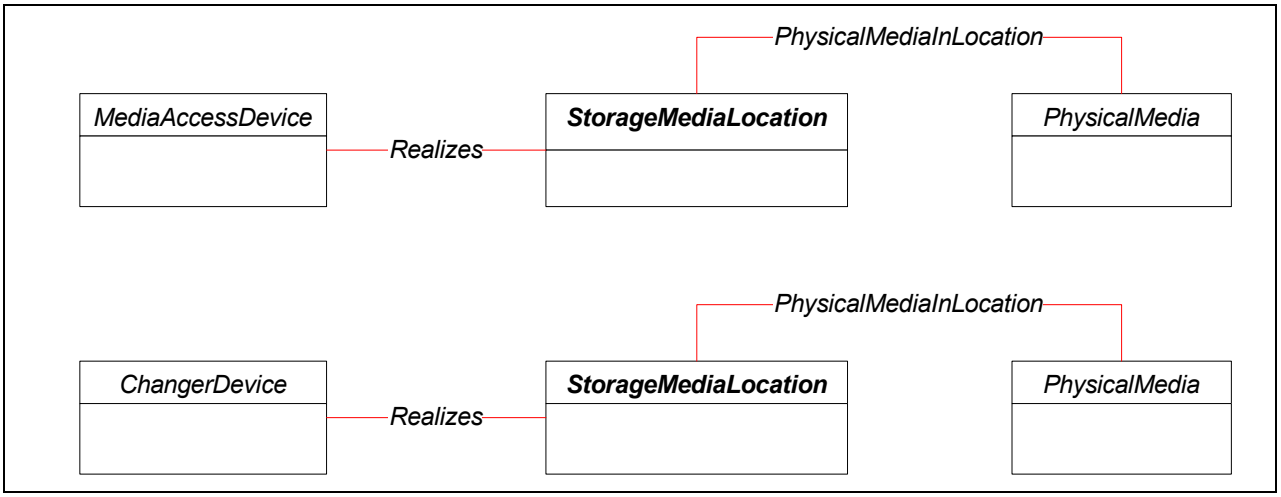

Figure 9 - StorageMediaLocation Instance Diagram

4.1.7 Durable Names and Correlatable IDs of the Profile

Different implementations use different approaches to uniquely identify the SCSI units pertinent to Storage Media Libraries (i.e., Changer Devices and Media Access Devices). The agent should utilize the same Durable Name techniques described for volumes in *Storage Management Technical Specification, Part 1 Common Architecture, 1.4.0 Rev 6*, 7.5 Guidelines for Storage System Names. The chosen name is stored in the Name attribute of the logical device with the corresponding setting for the NameFormat attribute. Allowable name formats and device pairings for the storage library profile are:

- FCPort: FCPort.PermanentAddress = Fibre Channel Port World Wide Name. NameFormat should be set to “WWN”
- ChangerDevice.DeviceID = Vendor+Product+Serial Number+(optional instance number). Vendor, Model and Serial number should be taken from the ChangerDevice’s associated ComputerSystem, Product, and/or Chassis. An option instance number may be added to uniquely denote more than one ChangerDevice “inside” a ComputerSystem
- MediaAccessDevice (or TapeDrive).DeviceID = Vendor+Product+Serial number for the MediaAccessDevice
- ComputerSystem.Name = Vendor+Product+Serial number for the storage library and/or its associated Product and Chassis. NameFormat should be set to “Vendor+Product+Serial”

Please refer to *Storage Management Technical Specification, Part 1 Common Architecture, 1.4.0 Rev 6* 7.6 for additional information.

4.2 Health and Fault Management Considerations

None

4.3 Cascading Considerations

None

4.4 Supported Subprofiles and Packages

Table 1 describes the supported profiles for Storage Library.

Table 1 - Supported Profiles for Storage Library

Profile Name	Organization	Version	Requirement	Description
Access Points	SNIA	1.3.0	Optional	
Location	SNIA	1.4.0	Optional	
FC Target Ports	SNIA	1.4.0	Optional	
Software	SNIA	1.4.0	Optional	
Storage Library Limited Access Port Elements	SNIA	1.2.0	Optional	
Storage Library Media Movement	SNIA	1.1.0	Optional	
Storage Library Capacity	SNIA	1.1.0	Optional	
Storage Library Element Counting	SNIA	1.1.0	Optional	
Storage Library InterLibraryPort Connection	SNIA	1.1.0	Optional	

Table 1 - Supported Profiles for Storage Library

Profile Name	Organization	Version	Requirement	Description
Storage Library Partitioned Library	SNIA	1.2.0	Optional	
Physical Package	SNIA	1.3.0	Mandatory	

4.5 Methods of this Profile

None

4.6 Client Considerations and Recipes

4.6.1 Recipe Overview

While no pseudo-code-based recipes have been written for this profile, this section provides some helpful information for writing management applications and suggests techniques for addressing common use cases.

4.6.2 Discover a Storage Media Library

Discovery of Storage Media Libraries is achieved by looking up instances of ComputerSystem which are subclassed from System and have a corresponding Name and NameFormat property as described above under 4.1.7. Specifically, NameFormat shall be set to “VendorModelSerial” and the Name shall be of the form Vendor+Product+Serial

4.6.3 Determine Library Physical Media Capacity

The physical media capacity of a library is the number of physical media objects that may be stored in the currently installed configuration of a Storage Media Library. This capacity may be determined by enumerating the StorageMediaLocation instances that are associated with each of the library’s Chassis objects.

In implementations that choose to include the Capacity subprofile, minimum and maximum slot capacities for a Storage Library are modeled in the ConfigurationCapacity, which is described in Clause 7: Library Capacity Subprofile. Since this use case relies on an optional part of the profile, it may not be supported by each agent implementation.

4.6.4 Determine Physical Media Inventory

To determine the physical media inventory of a storage library, clients should discover the Chassis instance associated with a particular ComputerSystem (via the ComputerSystemPackage association), and enumerate the PhyscialMedia instances associated with the Chassis through the PackagedComponent association.

4.6.5 Discover Storage Library Control Type

The control mechanism to a library is either one of these:

- SCSI Media Changer Commands directed to the library’s changer device
- Library control commands directed to a Library Control service

If a library does not have a ProtocolController instance associated via ProtocolControllerForUnit to the ChangerDevice then the client should conclude that an alternate mechanism for controlling the library is required. This mechanism may vary, but should be represented by an instance of a HostedService associated with the ComputerSystem which models the storage library.

4.6.6 Determine Library Drive Capacity

The current drive capacity of a library may be determined by enumerating the MediaAccessDevice instances through the SystemDevice association of the library.

When the optional Capacity subprofile is implemented, the number of drives discovered should be within the range indicated by the minimum and maximum capacity attribute found on the library Chassis' ElementCapacity association with ConfigurationCapacity for tape drives. This bounds check is not available if the Capacity subprofile is not implemented.

4.6.7 Determine Drive Data Path Technology

Clients can discover the data path protocol of each drive within a storage library by enumerating MediaAccessDevice instances, then following the ProtocolControllerForUnit association linking a MediaAccessDevivce with a ProtocolController. Properties within Contoller can then be queried for more information. If the MediaAccessDevice has a fibre channel interface, an FCPort instance is linked to its ProtocolController by a ProtocolControllerForPort association. See *Storage Management Technical Specification, Part 2 Common Profiles, 1.4.0 Rev 6* Clause 8: FC Target Ports Profile for more information on fibre channel connectivity.

4.6.8 Find asset Information

Information about the entire storage library is modeled in the Chassis instances associated with the ComputerSystem. Chassis properties include Manufacturer, Model, Version, and Tag. Tag is an arbitrary identifying string.

To identify asset information for the logical devices, a client should access the corresponding logical device through the ComputerSystem object's SystemDevice association. For each logical device instance the client may then check for asset information from the PhysicalElement associated through a Realizes association. Product information may also be available through the corresponding ProductPhysicalElement/ProductPhysicalComponent aggregation.

4.6.9 Discovery of Mailslots, Import/Export Elements or LimitedAccessPorts in a Storage Library

Clients may determine the number of LimitedAccessPorts in a library by enumerating the LimitedAccessPorts connected to a ComputerSystem instance via the SystemDevice association.

Note that some smaller libraries do not have the type of import/export element modeled by LimitedAccessPort. As a result, LimitedAccessPort elements are included in an (optional) subprofile (see Clause 9: Limited Access Port Elements Subprofile).

4.6.10 Counting assets in large storage libraries

Very large libraries may contain dozens of MediaAccessDevices and many thousands of StorageMediaLocations and PhysicalMedia. The intrinsic enumerateInstances() method is commonly used to count or gather CIM object instances of this type. Clients may find that using enumerateInstances() to count assets in very large libraries requires an excessive amount of time and processing resources. Providers supporting large libraries may also find that excessive time and resources are consumed attempting to return the bulk of data requested in enumerateInstances() calls. The following suggestions may be of help in situations where large libraries are of interest:

- Omit Qualifiers from enumerateInstances() or getInstance() requests;
- Request only the lowest-level child class of interest for examination or counting;
- Request only the properties of interest in enumerateInstances() or getInstance() requests. When only a count of existing objects is desired, omit all properties from the request;

- Use the intrinsic enumerateInstanceNames() or associatorNames() method instead of enumerateInstances() when only a count of existing objects is desired. The enumerateInstanceNames() and associatorNames() calls are much “lighter weight” overall than enumerateInstances();
- If the provider supports it, use the Physical Elements Count subprofile to quickly count PhysicalMedia and StorageMediaLocation instances. Note that this subprofile is optional and experimental and may not be supported by some providers.

4.7 Registered Name and Version

Storage Library version 1.2.0

4.8 CIM Elements

Table 2 describes the CIM elements for Storage Library.

Table 2 - CIM Elements for Storage Library

Element Name	Requirement	Description
4.8.1 CIM_ChangerDevice	Mandatory	
4.8.2 CIM_Chassis	Mandatory	
4.8.3 CIM_ComputerSystem	Mandatory	
4.8.4 CIM_ComputerSystem	Mandatory	'Top level' system that represents the whole Storage Library.
4.8.5 CIM_ComputerSystemPackage	Mandatory	
4.8.6 CIM_ElementCapabilities	Optional	Class to implement the association between the top-level ComputerSystem representing a Storage Library and it's StorageLibraryCapabilities.
4.8.7 CIM_ElementSoftwareIdentity	Mandatory	
4.8.8 CIM_ElementSoftwareIdentity	Mandatory	
4.8.9 CIM_MediaAccessDevice	Mandatory	
4.8.10 CIM_PackagedComponent	Mandatory	
4.8.11 CIM_PhysicalMedia	Mandatory	
4.8.12 CIM_PhysicalMediaInLocation	Mandatory	
4.8.13 CIM_ProtocolControllerForUnit	Mandatory	
4.8.14 CIM_Realizes	Conditional	Conditional requirement: Support for Inter-Library Port profile.
4.8.15 CIM_SCSIProtocolController	Mandatory	
4.8.16 CIM_SoftwareIdentity	Mandatory	

Table 2 - CIM Elements for Storage Library

Element Name	Requirement	Description
4.8.17 CIM_StorageLibraryCapabilities	Optional	Describes the capabilities of the Storage Library represented by the top level ComputerSystem this is associated with.
4.8.18 CIM_StorageMediaLocation	Mandatory	
4.8.19 CIM_SystemDevice	Conditional	Conditional requirement: Support for Inter-Library Port profile. This association links logicalDevices To the scoping system.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_ComputerSystem	Mandatory	Creation of a storage library instance.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_ComputerSystem	Mandatory	Deletion of a storage library instance.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_PhysicalMedia	Mandatory	Creation of a physical media instance.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_PhysicalMedia	Mandatory	Deletion of a physical media instance.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_MediaAccessDevice	Mandatory	Creation of a media access device instance.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_MediaAccessDevice	Mandatory	Deletion of a media access device instance.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_ChangerDevice	Mandatory	Creation of a Changer Device instance.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_ChangerDevice	Mandatory	Deletion of a Changer Device instance.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND PreviousInstance.OperationalStatus <> SourceInstance.OperationalStatus	Mandatory	Deprecated WQL -Change in OperationalStatus of a storage library.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_MediaAccessDevice AND PreviousInstance.OperationalStatus <> SourceInstance.OperationalStatus	Mandatory	Deprecated WQL -Change in OperationalStatus for a media access device.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ChangerDevice AND PreviousInstance.OperationalStatus <> SourceInstance.OperationalStatus	Mandatory	Deprecated WQL -Change in OperationalStatus for a Changer Device.

Table 2 - CIM Elements for Storage Library

Element Name	Requirement	Description
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND PreviousInstance.CIM_ComputerSystem::OperationalStatus <> SourceInstance.CIM_ComputerSystem::OperationalStatus	Mandatory	CQL -Change in OperationalStatus of a storage library.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_MediaAccessDevice AND PreviousInstance.CIM_MediaAccessDevice::OperationalStatus <> SourceInstance.CIM_MediaAccessDevice::OperationalStatus	Mandatory	CQL -Change in OperationalStatus for a media access device.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ChangerDevice AND PreviousInstance.CIM_ChangerDevice::OperationalStatus <> SourceInstance.CIM_ChangerDevice::OperationalStatus	Mandatory	CQL -Change in OperationalStatus for a Changer Device.

4.8.1 CIM_ChangerDevice

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

Table 3 describes class CIM_ChangerDevice.

Table 3 - SMI Referenced Properties/Methods for CIM_ChangerDevice

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
DeviceID		Mandatory	
MediaFlipSupported		Mandatory	
ElementName		Mandatory	

Table 3 - SMI Referenced Properties/Methods for CIM_ChangerDevice

Properties	Flags	Requirement	Description & Notes
OperationalStatus		Mandatory	Status of the changer device.
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.

4.8.2 CIM_Chassis

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

Table 4 describes class CIM_Chassis.

Table 4 - SMI Referenced Properties/Methods for CIM_Chassis

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Tag		Mandatory	
LockPresent		Mandatory	
SecurityBreach		Mandatory	
IsLocked		Mandatory	
ElementName		Mandatory	
Manufacturer		Mandatory	
Model		Mandatory	
SerialNumber		Mandatory	

4.8.3 CIM_ComputerSystem

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

Table 5 describes class CIM_ComputerSystem.

Table 5 - SMI Referenced Properties/Methods for CIM_ComputerSystem

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Name		Mandatory	

4.8.4 CIM_ComputerSystem

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

Table 6 describes class CIM_ComputerSystem.

Table 6 - SMI Referenced Properties/Methods for CIM_ComputerSystem

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Name		Mandatory	Unique identifier for the storage library. This should take the form of a string consisting of Vendor+Product+SerialNumber, derived from SCSI Inquiry Pages.
Dedicated		Mandatory	Indicates that this computer system is dedicated to operation as a storage library.
NameFormat		Mandatory	Format for Name property. **HID** is a required format. Others are optional.
OperationalStatus		Mandatory	Overall status of the library.
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.
ElementName		Mandatory	User friendly name.
PrimaryOwnerContact	M	Optional	Contact details for storage library owner.
PrimaryOwnerName	M	Optional	Owner of the storage library.

4.8.5 CIM_ComputerSystemPackage

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

Requirement: Mandatory

Table 7 describes class CIM_ComputerSystemPackage.

Table 7 - SMI Referenced Properties/Methods for CIM_ComputerSystemPackage

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

4.8.6 CIM_ElementCapabilities

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

Table 8 describes class CIM_ElementCapabilities.

Table 8 - SMI Referenced Properties/Methods for CIM_ElementCapabilities

Properties	Flags	Requirement	Description & Notes
Capabilities		Mandatory	
ManagedElement		Mandatory	

4.8.7 CIM_ElementSoftwareIdentity

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

Table 9 describes class CIM_ElementSoftwareIdentity.

Table 9 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

4.8.8 CIM_ElementSoftwareIdentity

Created By: Static

Modified By: Static
Deleted By: Static
Requirement: Mandatory

Table 10 describes class CIM_ElementSoftwareIdentity.

Table 10 - SMI Referenced Properties/Methods for CIM_ElementSoftwareIdentity

Properties	Flags	Requirement	Description & Notes
Dependent		Mandatory	
Antecedent		Mandatory	

4.8.9 CIM_MediaAccessDevice

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

Table 11 describes class CIM_MediaAccessDevice.

Table 11 - SMI Referenced Properties/Methods for CIM_MediaAccessDevice

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
DeviceID		Mandatory	
OperationalStatus		Mandatory	
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.
NeedsCleaning		Mandatory	If unknown, set to False.
MountCount		Mandatory	

4.8.10 CIM_PackagedComponent

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

Table 12 describes class CIM_PackagedComponent.

Table 12 - SMI Referenced Properties/Methods for CIM_PackagedComponent

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

4.8.11 CIM_PhysicalMedia

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

Table 13 describes class CIM_PhysicalMedia.

Table 13 - SMI Referenced Properties/Methods for CIM_PhysicalMedia

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Tag		Mandatory	
Capacity		Mandatory	0 = unknown. If CleanerMedia=True, then ignore Capacity value.
MediaType		Mandatory	
MediaDescription		Optional	
CleanerMedia		Mandatory	If unknown, set to False.
DualSided		Mandatory	
LabelStates		Mandatory	
LabelFormats		Mandatory	
PhysicalLabels		Mandatory	
RemovalConditions		Mandatory	

4.8.12 CIM_PhysicalMediaInLocation

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

Table 14 describes class CIM_PhysicalMediaInLocation.

Table 14 - SMI Referenced Properties/Methods for CIM_PhysicalMediaInLocation

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

4.8.13 CIM_ProtocolControllerForUnit

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

Table 15 describes class CIM_ProtocolControllerForUnit.

Table 15 - SMI Referenced Properties/Methods for CIM_ProtocolControllerForUnit

Properties	Flags	Requirement	Description & Notes
DeviceNumber		Optional	The target device visible through the controller.
Antecedent		Mandatory	Reference to MediaAccessDevice or ChangerDevice.
Dependent		Mandatory	

4.8.14 CIM_Realizes

Created By: Static
Modified By: Static
Deleted By: Static
Requirement: Support for Inter-Library Port profile.

Table 16 describes class CIM_Realizes.

Table 16 - SMI Referenced Properties/Methods for CIM_Realizes

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

4.8.15 CIM_SCSIProtocolController

This is only required if FC Ports claim backwards compatibility with SMI-S 1.0.

Created By: Static
Modified By: Static

Deleted By: Static
Requirement: Mandatory

Table 17 describes class CIM_SCSIProtocolController.

Table 17 - SMI Referenced Properties/Methods for CIM_SCSIProtocolController

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
DeviceID		Mandatory	Opaque identifier.
ElementName		Optional	
OperationalStatus		Mandatory	
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.
MaxUnitsControlled		Optional	

4.8.16 CIM_SoftwareIdentity

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

Table 18 describes class CIM_SoftwareIdentity.

Table 18 - SMI Referenced Properties/Methods for CIM_SoftwareIdentity

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
VersionString		Mandatory	The software of firmware version of the device (ChangerDevice, MediaAccessDevice, or a SCSIProtocolController).
Manufacturer		Mandatory	
Classifications		Optional	4 = Application Software, 10 = Firmware.
BuildNumber		Optional	
MajorVersion		Optional	
RevisionNumber		Optional	
MinorVersion		Optional	

4.8.17 CIM_StorageLibraryCapabilities

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

Table 19 describes class CIM_StorageLibraryCapabilities.

Table 19 - SMI Referenced Properties/Methods for CIM_StorageLibraryCapabilities

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	Unique Identifier for this Capabilities class. See MOF for specific format.
ElementName		Mandatory	A user friendly name.
Capabilities		Optional	Array of general capabilities for the Storage Library (see MOF).
MaxAuditTime		Optional	Number of seconds it takes for the library to complete an audit or "inventory" operations.

4.8.18 CIM_StorageMediaLocation

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

Table 20 describes class CIM_StorageMediaLocation.

Table 20 - SMI Referenced Properties/Methods for CIM_StorageMediaLocation

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Tag		Mandatory	
LocationType		Mandatory	
LocationCoordinates		Mandatory	
MediaTypesSupported		Mandatory	
MediaCapacity		Mandatory	

4.8.19 CIM_SystemDevice

Created By: Static
Modified By: Static
Deleted By: Static
Requirement: Support for Inter-Library Port profile.

Table 21 describes class CIM_SystemDevice.

Table 21 - SMI Referenced Properties/Methods for CIM_SystemDevice

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

STABLE

EXPERIMENTAL

Clause 5: Element Counting Subprofile

5.1 Description

The Element counting subprofile defines methods to count the number of physical tapes, storage media locations, and other classes within a storage library (or other system type). Such methods allow clients to avoid retrieving all *instances* of physical element classes simply to count them. Therefore, network traffic will be saved between client applications and storage library providers. These methods are modeled by the ConfigurationReportingService hosted by the storage library's (or other system type's) top-level ComputerSystem.

Figure 10 provides a sample instance diagram.

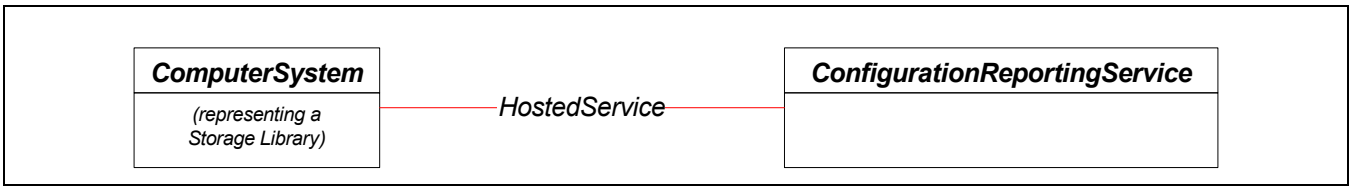

Figure 10 - Instance Diagram

5.1.1 Discovery

The Element counting subprofile, as currently defined, is not an advertised profile. Support for the Element Counting Subprofile can be obtained through the Storage Library Profile (or other top-level system profile as appropriate).

5.2 Health and Fault Management Considerations

Not defined in this standard.

5.3 Cascading Considerations

Not defined in this standard.

5.4 Supported Subprofiles and Packages

Related Profiles for Storage Library Element Counting: Not defined in this standard.

5.5 Methods of the Profile

5.5.1 GetClassTypes

GetClassTypes returns the list of class types that a given ManagedElement – typically, a storage library's top-level ComputerSystem or Chassis – supports or has installed. Calling GetClassTypes in the first step in a three step process to obtain a count of desired elements. (See 5.6 Client Considerations and Recipes for an overview and example).

The GetClassTypes method uses the following parameters:

[IN] uint16 InquiryType = “Installed” or “Supports”

When “Installed” is specified, the method will return the list of countable classes that the associated ComputerSystem currently has installed or contained within its scope. When “Supports” is specified, the method will return the list of countable classes that the associated ComputerSystem potentially supports, though no such class instances may currently be installed or contained within its scope.

[IN] boolean Recursive = true or false

For the purposes of the current subprofile, the value of the Recursive parameter is not relevant. Until defined otherwise, clients should specify “false”, and expect that the value will not affect operation of the GetClassTypes method in any way.

[IN] CIM_ManagemedElement REF Target = a CIM object pointer to the to the top-level ComputerSystem to which ConfigurationReportingService is associated. In some cases, a pointer the ComputerSystem’s Chassis may be appropriate. This parameter reinforces that the ConfigurationReportingService is returning information on the storage library’s (or other top-level profile’s) ComputerSystem or Chassis. Classes to be returned or counted are considered to be uniquely within the scope of this top-level ComputerSystem or Chassis.

[IN (false], OUT] string ClassTypes[] = an array of class types that can be counted by the service. One value of this parameter will be selected by the client and used when calling GetUnitTypes() and ReportCapacity(), described below. The method/service provider may return a string representation of any valid CIM class which it can report a count on. For example, a storage library provider might return “CIM_PhysicalMedia” to indicate that this service allows clients to obtain a count of PhysicalMedia instances currently associated with the Target ComputerSystem or Chassis instance. Other example values would be “CIM_StorageMediaLocation” and “CIM_MediaAccessDevice”

The GetClassTypes method also returns one of the following status values:

“Success”, “Not Supported”, “Unknown”, “Timeout”, “Failed”, “DMTF Reserved”, “Vendor Specific”. In general, it is expected that “Success” will be returned on successful execution and “Failed” or “Timeout” will be returned when errors occur in executing this method on the provider/server side. If “Not Supported” is returned, the client may still attempt to call the GetUnitTypes and ReportCapacity methods, but a known value for the ClassType parameter will not be available to the client up front. “Unknown” indicates that the result cannot be determined for the given parameter combination at this time.

5.5.2 GetUnitTypes

GetUnitTypes returns the type of “unit” relationships that can be specified by the client when counting class instances associated with a top-level ComputerSystem or Chassis. Calling GetUnitTypes in the second step in a three step process to obtain a count of desired elements. (See 5.6 Client Considerations and Recipes for an overview and example).

The GetUnitTypes method uses many of the same parameters as GetClassTypes, including:

[IN] uint16 InquiryType: see details in 5.5.1 GetClassTypes. “Supported” or “Installed” are valid enumerated values.

[IN] boolean Recursive: see details in 5.5.1 GetClassTypes. Generally, a value of “false” is expected.

[IN] CIM_ManagedElement REF Target: see details in 5.5.1 GetClassTypes. A pointer to the top-level ComputerSystem associated with this ConfigurationReportingService. In some cases, a pointer to the top-level Chassis may be appropriate.

[IN] string ClassType: see details in 5.5.1 GetClassTypes. The class type to be counted.

[IN (false) OUT] uint16 UnitTypes[] = an array of “relationship types” to help specify how the class instances to be counted are associated with the top-level ComputerSystem or Chassis specified by Target. Many values are available for UnitTypes, but clients should expect that only “Contained” or “Connected” will be

returned by storage library providers. Other values, such as “None”, “Front Side”, and “Memory” should not be returned until future definition of their meaning is documented. Clients will use one of the values returned in this parameter when calling ReportCapacity.

The GetUnitTypes method also returns one of the following status values:

“Success”, “Not Supported”, “Unknown”, “Timeout”, “Failed”, “DMTF Reserved”, “Vendor Specific”. In general, it is expected that “Success” will be returned on successful execution and “Failed” or “Timeout” will be returned when errors occur in executing this method on the provider/server side. If “Not Supported” is returned, the client may still attempt to call the ReportCapacity method, but a known value for the UnitType parameter will not be available to the client up front. In general, clients should attempt to specify “Contained” or “Connected” when calling ReportCapacity. “Unknown” indicates that the result cannot be determined for the given parameter combination at this time.

5.5.3 ReportCapacity

ReportCapacity returns the number or count of a given class types that the given ManagementElement – typically, a storage library’s top-level ComputerSystem or Chassis – supports or has installed. Calling ReportCapacity in the third step in a three step process to obtain a count of desired elements. (See 5.6 Client Considerations and Recipes for an overview and example).

The ReportCapacity method uses many of the same parameters as GetClassTypes and GetUnitTypes, including:

[IN] uint16 InquiryType: see details in 5.5.1 GetClassTypes. “Supported” or “Installed” are valid enumerated values.

[IN] boolean Recursive: see details in 5.5.1 GetClassTypes. Generally, a value of “false” is expected.

[IN] CIM_ManagedElement REF Target: in 5.5.1 GetClassTypes. A pointer to the top-level ComputerSystem associated with this ConfigurationReportingService. In some cases, a pointer to the top-level Chassis may be appropriate.

[IN] string ClassType: see details in 5.5.1 GetClassTypes. The class type to be counted.

[IN] uint16 UnitType: see details in 5.5.1 GetClassTypes. Generally, the “Contained” or “Connected” enumerated value will be used.

[IN (false), OUT] uint64 NumberOfUnits = the number of “supported” or “installed” ClassType instances “contained” or “connected” in a given Target ComputerSystem’s (or Chassis’s) scope. Obtaining this count is the purpose of the ConfigurationReportingService.

The ReportCapacity method also returns one of the following status values:

“Success”, “Not Supported”, “Unknown”, “Timeout”, “Failed”, “DMTF Reserved”, “Vendor Specific”. In general, it is expected that “Success” will be returned on successful execution and “Failed” or “Timeout” will be returned when errors occur in executing this method on the provider/server side. If “Not Supported” is returned, it may indicate that the Target, ClassType, or UnitType parameters are in error. Supported values for ClassType and UnitType should be obtained by calling GetClassTypes and GetUnitTypes prior to calling ReportCapacity. “Unknown” indicates that the result cannot be determined for the given parameter combination at this time.

5.6 Client Considerations and Recipes

ConfigurationReportingService may be used by clients interested in quickly obtaining a count or “number of” desired instances. For example, a client may want to know the number of PhysicalMedia instances associated with a particular storage library, but the time and overhead associated with enumerating the instances of these objects – through the extrinsic enumerateInstances() or enumerateInstanceNames() methods – can be excessive.

To use ConfigurationReportingService, clients call three methods in succession: GetClassTypes, GetUnitTypes, and ReportCapacity. GetClassTypes returns the list of class types that can be counted. This information is then

used to call GetUnitTypes, which returns a list of “unit” relationships (e.g. “Connected” or “Contained”). This value and other information is then passed to ReportCapacity, which returns the count of desired class instances.

An example: A client wants to count the number of PhysicalMedia instances associated with a storage library (itself represented by a top-level ComputerSystem and Chassis instance). Having discovered a ConfigurationReportingService associated with the ComputerSystem of interest, the client will call:

```
uint32 GetClassTypes (
     InquiryType = “Installed”,
     Recursive = “false”,
     Target = CIM object path to the ComputerSystem of interest,
     &ClassTypes[] = pointer to the countable classes, as returned by the
                          provider/service)
```

Assuming that GetClassTypes returns a value of “Success”, the client may examine the ClassTypes[] array and find that it contains “CIM_MediaAccessDevice”, “CIM_PhysicalMedia”, “CIM_StorageMediaLocation”, and “CIM_MediaTranferDevice”. Since this client is interested in PhysicalMedia, it would use the “CIM_PhysicalMedia” value use to call GetUnitTypes:

```
uint32 GetUnitTypes (
     InquiryType = “Installed”,
     Recursive = “false”,
     Target = CIM object path to the ComputerSystem of interest,
     ClassType = “CIM_PhysicalMedia”
     &UnitTypes[] = pointer to the supported “unit” relationship types, as
                          returned by the provider/service)
```

Assuming that GetUnitTypes returns a value of “Success”, the client may examine the UnitTypes[] array and find that it contains only “Contained”. The client would then use this value to call ReportCapacity:

```
uint32 ReportCapacity (
     InquiryType = “Installed”,
     Recursive = “false”,
     Target = CIM object path to the ComputerSystem of interest,
     ClassType = “CIM_PhysicalMedia”,
     UnitType = “Contained”
     &NumberOfUnits)
```

Assuming that ReportCapacity returns a value of “Success”, the client should examine the NumberOfUnits value to determine the number of CIM_PhysicalMedia “contained” or currently “installed” in the Target ComputerSystem.

In general, it is expected that “Success” will be returned on successful execution of these three methods, and “Failed” or “Timeout” will be returned when errors occur in executing these methods on the provider/server side. If “Not Supported” is returned, it may indicate that the Target, ClassType, or UnitType parameters are in error.

5.7 Registered Name and Version

Storage Library Element Counting version 1.1.0

5.8 CIM Elements

Table 22 describes the CIM elements for Storage Library Element Counting.

Table 22 - CIM Elements for Storage Library Element Counting

Element Name	Requirement	Description
5.8.1 CIM_ConfigurationReportingService	Mandatory	
5.8.2 CIM_HostedService	Mandatory	

5.8.1 CIM_ConfigurationReportingService

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

Table 23 describes class CIM_ConfigurationReportingService.

Table 23 - SMI Referenced Properties/Methods for CIM_ConfigurationReportingService

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
Name		Mandatory	
GetClassTypes()		Mandatory	
GetUnitTypes()		Mandatory	
ReportCapacity()		Mandatory	

5.8.2 CIM_HostedService

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

Table 24 describes class CIM_HostedService.

Table 24 - SMI Referenced Properties/Methods for CIM_HostedService

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

EXPERIMENTAL

EXPERIMENTAL

Clause 6: InterLibraryPort Connection Subprofile

6.1 Description

Support of InterLibraryPort devices, a.k.a. pass-thru ports or cartridge exchange mechanisms, is designated as optional in this profile. However, when such a device exists the agent representing the library should instantiate this class for each port. When one or more libraries are connected via an Inter-Library Port and the corresponding agents are working with separate name spaces a mechanism is required for correlating the LibraryExchange association that represents the port connection.

Figure 11 provides a sample instance diagram.

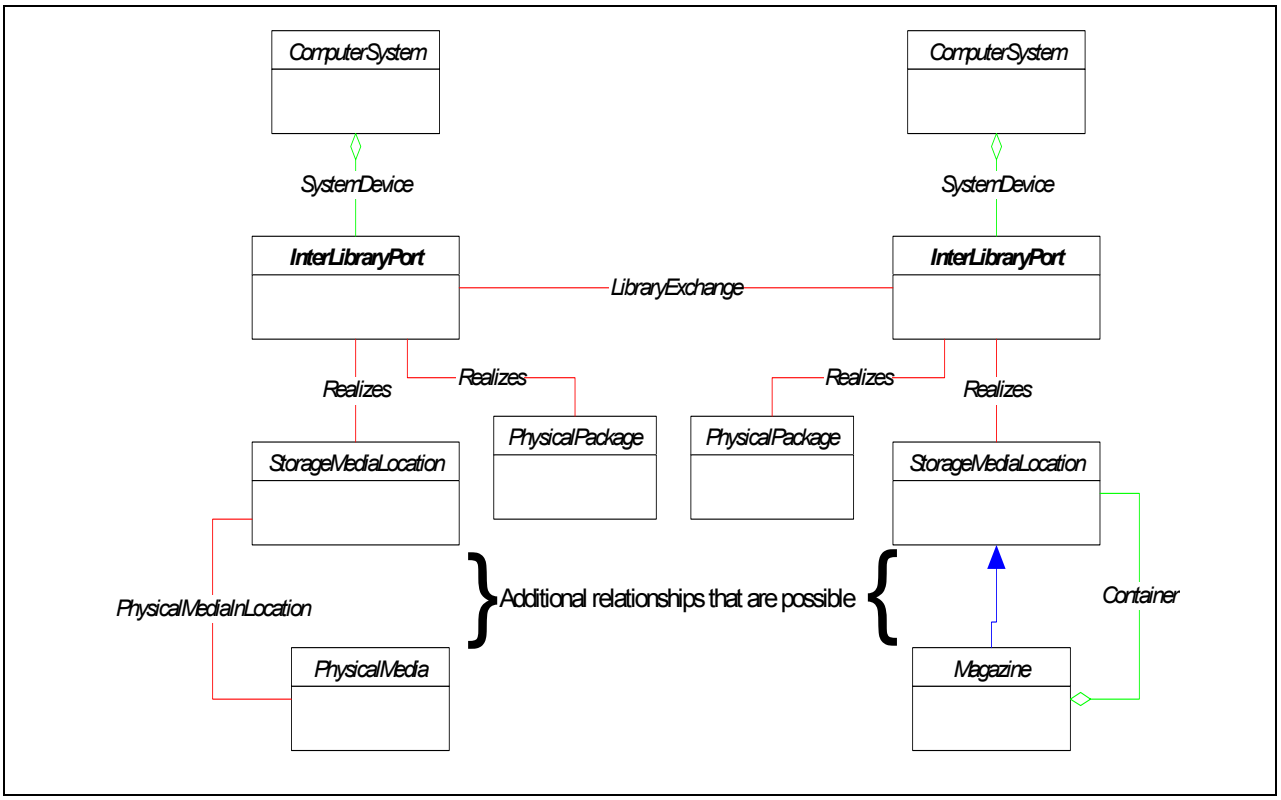

Figure 11 - InterLibraryPort Connection Instance Diagram

Durable Names and Correlatable IDs

A Durable Name is not defined by this profile for InterLibraryPort instances and remains unspecified. This is not an issue when associated InterLibraryPort instances are within the same name space.

6.2 Health and Fault Management Considerations

Not defined in this standard.

6.3 Cascading Considerations

Not defined in this standard.

6.4 Supported Subprofiles and Packages

None.

6.5 Methods of the Profile

None.

6.6 Client Considerations and Recipes

None.

6.7 Registered Name and Version

Storage Library InterLibraryPort Connection version 1.1.0

6.8 CIM Elements

Table 25 describes the CIM elements for Storage Library InterLibraryPort Connection.

Table 25 - CIM Elements for Storage Library InterLibraryPort Connection

Element Name	Requirement	Description
6.8.1 CIM_InterLibraryPort	Mandatory	InterLibraryPorts represent hardware that transports Physical Media between connected Storage Libraries. The LibraryExchange association identifies the connected Libraries, by identifying the connected InterLibraryPorts.
6.8.2 CIM_LibraryExchange	Mandatory	This relationship identifies that two storage libraries are connected through their InterLibraryPorts.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_InterLibraryPort	Mandatory	Creation of an instance of InterLibraryPort.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_InterLibraryPort	Mandatory	Deletion of an instance of InterLibraryPort.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_InterLibraryPort AND SourceInstance.OperationalStatus <> PreviousInstance.OperationalStatus	Mandatory	Deprecated WQL -Change in OperationalStatus of a InterLibraryPort.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_InterLibraryPort AND SourceInstance.CIM_InterLibraryPort::OperationalStatus <> PreviousInstance.CIM_InterLibraryPort::OperationalStatus	Mandatory	CQL -Change in OperationalStatus of a InterLibraryPort.

6.8.1 CIM_InterLibraryPort

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

Table 26 describes class CIM_InterLibraryPort.

Table 26 - SMI Referenced Properties/Methods for CIM_InterLibraryPort

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
DeviceID		Mandatory	
LastAccessed		Mandatory	Last access time of the port by the library.
ImportCount		Mandatory	The number of times the port was used to move physical media into the storage library.
ExportCount		Mandatory	The number of times the port was used to move physical media out of the storage library.
Direction		Mandatory	Identifies whether the port can be used to import physical media, export physical media or both.
OperationalStatus		Mandatory	Status of the InterLibrary port.
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.

6.8.2 CIM_LibraryExchange

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

Table 27 describes class CIM_LibraryExchange.

Table 27 - SMI Referenced Properties/Methods for CIM_LibraryExchange

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

EXPERIMENTAL

EXPERIMENTAL

Clause 7: Library Capacity Subprofile

7.1 Description

By adding two classes (ConfigurationCapacity and ElementCapacity) servers can publish the minimum and maximum number of slots, drives, magazines, media changers, and other elements associated with a given storage library.

Figure 12 illustrates the use of ConfigurationCapacity and ElementCapacity in conjunction with the basic storage library profile.

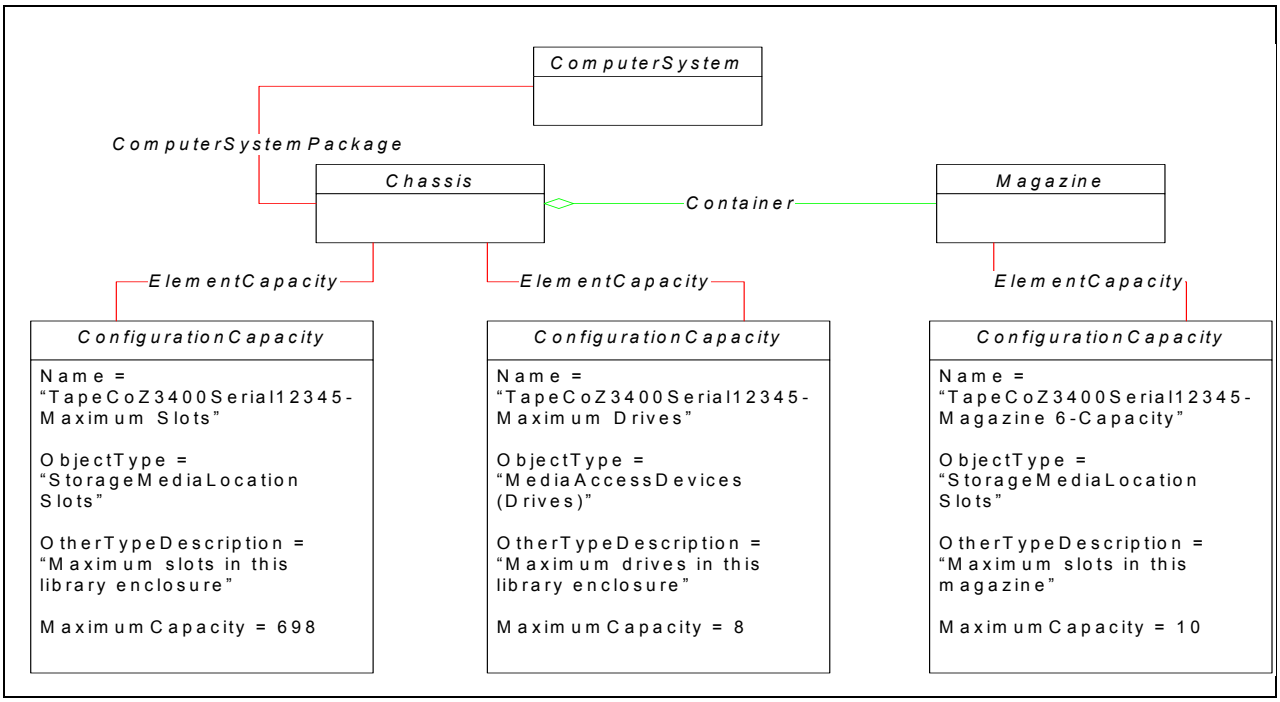

Figure 12 - Library Capacity Instance Diagram

7.2 Health and Fault Management Considerations

Not defined in this standard.

7.3 Cascading Considerations

Not defined in this standard.

7.4 Supported Subprofiles and Packages

None.

7.5 Client Considerations and Recipes

None.

7.6 Registered Name and Version

Storage Library Capacity version 1.1.0

7.7 CIM Elements

Table 28 describes the CIM elements for Storage Library Capacity.

Table 28 - CIM Elements for Storage Library Capacity

Element Name	Requirement	Description
7.7.1 CIM_ConfigurationCapacity	Mandatory	ConfigurationCapacity provides information on the minimum and maximum number of slots, drives, magazines, media changers, and other elements associated with a given storage library.
7.7.2 CIM_ElementCapacity	Mandatory	

7.7.1 CIM_ConfigurationCapacity

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

Table 29 describes class CIM_ConfigurationCapacity.

Table 29 - SMI Referenced Properties/Methods for CIM_ConfigurationCapacity

Properties	Flags	Requirement	Description & Notes
Name		Mandatory	
ObjectType		Mandatory	Other, Processors, Power Supplies, see MOF.
OtherTypeDescription		Optional	
MinimumCapacity		Mandatory	
MaximumCapacity		Mandatory	

7.7.2 CIM_ElementCapacity

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

Table 30 describes class CIM_ElementCapacity.

Table 30 - SMI Referenced Properties/Methods for CIM_ElementCapacity

Properties	Flags	Requirement	Description & Notes
Element		Mandatory	
Capacity		Mandatory	

EXPERIMENTAL

EXPERIMENTAL

Clause 8: LibraryAlert Events/Indications for Library Devices

8.1 Description

Historically, media libraries have been managed using both SCSI and SNMP interfaces. A number of library management standards have been defined based on these interfaces, including the “TapeAlert” error events flags. These events alert subscribing clients to current or pending error conditions related to a library, drives, or media. The SCSI implementation of TapeAlert is described in the SCSI Stream Commands (SSC-2) and SCSI Media Changer Commands (SMC-2) specifications.

In order to carry these useful asynchronous events into the WBEM/CIM domain, the TapeAlert events have been mapped into instances of the AlertIndication class. This CIM class provides a general means for communicating asynchronous events to subscribing clients and TapeAlert events/indications -- hereafter referred to more generally as “LibraryAlert” indications -- shall be specified by filling in standard values for the properties of an AlertIndication.

8.2 Health and Fault Management Considerations

Not defined in this standard.

8.3 Cascading Considerations

Not defined in this standard.

8.4 Supported Subprofiles and Packages

None.

8.5 Methods of the Profile

None.

8.6 Client Considerations and Recipes

For all LibraryAlert indications, the following properties of AlertIndication shall be static and set to the values shown in Table 31.

Table 31 - LibraryAlert Property Settings

Property Name	Property type	Property Value
Description	string	“LibraryAlert Indication”
AlertType	Uint16 (enumeration)	5 = “Device Alert”
ProbableCause	Uint16 (enumeration)	1 = “other”
Trending	Uint16 (enumeration)	1 = “Not Applicable”
SystemCreationClassName	string	“CIM_ComputerSystem”

Clients may identify a received AlertIndication as a LibraryAlert indication primarily by the value of “LibraryAlert Indication” in the Description property. The following Query attribute on an IndicationFilter instance should be provided by the agent for these alerts:

```
SELECT * FROM CIM_Alert
                                WHERE Description=”LibraryAlert Indication”
```

The following AlertIndication properties for LibraryAlert indications shall be vendor-specific and no specification or restriction of values is made here:

Table 32 - Vendor Specific Properties of LibraryAlert

Property Name	Property type	Property Value
OtherSeverity	string	specified by vendor
EventID	string	specified by vendor
ProviderName	string	specified by vendor

A small number of AlertIndication properties for LibraryAlert indications shall have variable values that are restricted within a small range, as follows:

Table 33 - Variable Alert Properties for LibraryAlert

Property Name	Property type	Property Value
SystemName	string	Name property value for the StorageLibrary instance that is associated with this unique indication
AlertingManagedElement	string	CIMInstance in string format for element to which this indication applies: MediaAccessDevice, StorageLibrary, or PhysicalMedia

The remaining AlertIndication properties for LibraryAlert indications shall have values derived from the SCSI TapeAlert specifications: SCSI Stream Commands (SSC-2) and SCSI Media Changer Commands (SMC-2).

Note that a small number of indications apply only to Tape libraries, while all other indications apply generically to any library type. Those indications that are tape-specific may be identified by the following strings in the OtherAlertType property:

Table 34 - SCSI TapeAlert-based Properties

Property Name	Property type	Property Value
OtherAlertType	string	“Tape snapped/cut in the drive where media can be de-mounted.”
OtherAlertType	string	“Tape snapped/cut in the drive where media cannot be de-mounted.”
OtherAlertType	string	“The drive is having severe trouble reading or writing, which will be resolved by a retention cycle.”

The remaining AlertIndication properties and values for all LibraryAlert indications are shown in Table 35. Note that the OtherAlertType property, in particular, serves to uniquely identify each of the LibraryAlert indications.

Table 35 - LibraryAlert AlertIndication Properties

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Read Warning	"The drive is having severe trouble reading."	"3" = "Degraded/ Warning"	"The drive is having problems reading data. No data has been lost, but there has been a reduction in the performance."	
Write Warning	"The drive is having severe trouble writing."	"4" = "Warning"	"Worn out Media"	"1. Discard the worn out media." "2. Use a new cleaning media."
Hard Error	"The drive had a hard read or write error."	"5" = "Warning"	"Bad Media or Drive. The operation has stopped because an error has occurred while reading or writing data that the drive cannot correct."	
Media	"Media can no longer be written/read, or performance is severely degraded."	"6" = "Critical"	"Bad Media"	"1. Copy any data you require from this media." "2. Do not use this media again." "3. Restart the operation with a different media."
Read Failure	"The drive can no longer read data from the storage media."	"6" = "Critical"	"Worn out media"	"1. Replace media." "2. Call the drive supplier help line."
Write Failure	"The drive can no longer write data to the media."	"6" = "Critical"	"The media is from a faulty batch or the drive is faulty: "	"1. Use known-good media to test the drive. " "2. If the problem persists, call the media drive supplier"

Table 35 - LibraryAlert AlertIndication Properties (Continued)

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Media Life	"The media has exceeded its specified life."	"3" = "Degraded/ Warning"	"The media has reached the end of its calculated useful life:"	"1. Copy any data you need to another media." 2. Discard the old media."
Not Data Grade	"The cartridge is not data-grade. Any data you write to the media is at risk. Replace the cartridge with a data-grade media."	"3" = "Degraded/ Warning"	"The cartridge is not data-grade. Any data you write to the media is at risk."	"Replace the cartridge with a data-grade media."
Write Protect	"Write command is attempted to a write protected media."	"6" = "Critical"	"Replace with writable media"	"You are trying to write to a write protected cartridge. Remove the write protection or use another media."
No Removal	"Manual or software unload attempted when prevent media removal is on."	"2" = "Information"	"Wait until drive is not in-use."	"You cannot eject the cartridge because the drive is in use. Wait until the operation is complete before ejecting the cartridge."
Cleaning Media	"Cleaning media loaded into drive"	"2" = "Information"	"The media in the drive is a cleaning cartridge."	"Replace this media with writeable media"
Unsupported Format	"Attempted load of unsupported media format (e.g., DDS2 in DDS1 drive)."	"2" = "Information"	"You have tried to load a cartridge of a type that is not supported by this drive."	"Insert media of a type supported by this drive"
Recoverable Snapped Tape	"Tape snapped/cut in the drive where media can be de-mounted."	"6" = "Critical"	"The operation has failed because the tape in the drive has snapped:"	"1. Discard the old tape." "2. Restart the operation with a different tape."

Table 35 - LibraryAlert AlertIndication Properties (Continued)

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Unrecoverable Snapped Tape	"Tape snapped/cut in the drive where media cannot be de-mounted."	"6" = "Critical"	"The operation has failed because the tape in the drive has snapped:"	"1. Do not attempt to extract the tape cartridge." "2. Call the tape drive supplier help line."
Memory Chip In Cartridge Failure	"Memory chip failed in cartridge."	"3" = "Degraded/ Warning"	"The memory in the media has failed, which reduces performance.	"Do not use the cartridge for further write operations."
Forced Eject	"Manual or forced eject while drive actively writing or reading."	"6" = "Critical"	"The operation has failed because the media was manually de-mounted while the drive was actively writing or reading."	
Read Only Format	"Media loaded that is read-only format."	"3" = "Degraded/ Warning"	"You have loaded a cartridge of a type that is read-only in this drive. The cartridge will appear as write protected."	
Directory Corrupted On Load	"Drive powered down while loaded, or permanent error prevented the directory being updated."	"3" = "Degraded/ Warning"	"The directory on the cartridge has been corrupted. File search performance will be degraded. "	"The directory can be rebuilt by reading all the data on the cartridge."
Nearing Media Life	"Media may have exceeded its specified number of passes."	"2" = "Information"	"The storage media is nearing the end of its calculated life."	"1. Use another storage media for your next backup. "2. Store this storage media in a safe place in case you need to restore data from it."

Table 35 - LibraryAlert AlertIndication Properties (Continued)

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Clean Now	"The drive thinks it has a head clog or needs cleaning."	"6" = "Critical"	"The drive needs cleaning:"	"1. If the operation has stopped, eject the storage media and clean the drive." "2. If the operation has not stopped, wait for it to finish and then clean the drive. Check the drive user's manual for device specific cleaning."
Clean Periodic	"The drive is ready for a periodic cleaning."	"3" = "Degraded/ Warning"	"The drive is due for routine cleaning:"	"1. Wait for the current operation to finish." "2. Then use a cleaning cartridge. Check the drive user's manual for device specific cleaning instructions."
Expired Cleaning Media	"The cleaning media has expired."	"6" = "Critical"	"The last cleaning cartridge used in the drive has worn out:"	"1. Discard the worn out cleaning cartridge." "2. Wait for the current operation to finish." "3. Then use a new cleaning cartridge."
Invalid Cleaning Media	"Invalid cleaning media type used."	"6" = "Critical"	"The last cleaning cartridge used in the drive was an invalid type:"	"1. Do not use this cleaning cartridge in this drive." "2. Wait for the current operation to finish." "3. Then use a valid cleaning cartridge."

Table 35 - LibraryAlert AlertIndication Properties (Continued)

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Retention Requested	"The drive is having severe trouble reading or writing, which will be resolved by a retention cycle."	"3" = "Information"	"The drive has requested a retention operation."	
Dual-Port Interface Error	"Failure of one interface port in a dual-port configuration (i.e., Fibre Channel)"	"3" = "Degraded/ Warning"	"A redundant interface port on the drive has failed."	
Cooling Fan Failure	"Fan failure inside drive mechanism or drive enclosure."	"3" = "Degraded/ Warning"	"A drive cooling fan has failed."	"Replace cooling fan or drive enclosure"
Power Supply Failure	"Redundant power supply unit failure inside the drive enclosure or rack subsystem."	"3" = "Degraded/ Warning"	"A redundant power supply has failed inside the drive enclosure."	"Check the enclosure user's manual for instructions on replacing the failed power supply."
Power Consumption	"Power consumption of the drive is outside specified range."	"3" = "Degraded/ Warning"	"The drive power consumption is outside the specified range."	
Drive Maintenance	"The drive requires preventive maintenance (not cleaning)."	"3" = "Degraded/ Warning"	"Preventive maintenance of the drive is required."	"Check the drive users manual for device specific preventive maintenance tasks or call the drive supplier help line."
Hardware A	"The drive has a hardware fault that requires reset to recover."	"6" = "Critical"	"The drive has a hardware fault"	"1. Eject the media or magazine." "2. Reset the drive." "3. Restart the operation."

Table 35 - LibraryAlert AlertIndication Properties (Continued)

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Hardware B	"The drive has a hardware fault that is not read/write related or requires a power cycle to recover."	"6" = "Critical"	"The drive has a hardware fault"	"1. Turn the drive off and then on again." "2. Restart the operation." "3. If the problem persists, call the drive supplier help line."
Interface	"The drive has identified an interface fault."	"3" = "Degraded/ Warning"	"Bad cable or drive interface."	"1. Check the cables and cable connections." "2. Restart the operation."
Eject Media	"Error recovery action: Media Ejected"	"6" = "Critical"		"1. Eject the media or magazine." "2. Insert the media or magazine again." "3. Restart the operation."
Download Failure	"Firmware download failed."	"3" = "Degraded/ Warning"	"The firmware download has failed because you have tried to use the incorrect firmware for this drive."	"Obtain the correct firmware and try again."
Drive Humidity	"Drive humidity limits exceeded."	"3" = "Degraded/ Warning"	"Bad drive fan"	"Replace fan or drive enclosure"
Drive Temperature	"Drive temperature limits exceeded."	"3" = "Degraded/ Warning"	"Bad cooling fan"	"Replace fan or drive enclosure"
Drive Voltage	"Drive voltage limits exceeded."	"3" = "Degraded/ Warning"	"Bad drive power supply"	"Check the drive users manual for device specific preventive maintenance tasks or call the drive supplier help line."
Predictive Failure	"Predictive failure of drive hardware."	"6" = "Critical"		"A hardware failure of the drive is predicted. Call the drive supplier help line."

Table 35 - LibraryAlert AlertIndication Properties (Continued)

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Diagnostics Required	"The drive may have a hardware fault that may be identified by extended diagnostics (i.e., SEND DIAGNOSTIC command)."	"3" = "Degrading/ Warning"	"The drive may have a hardware fault."	"Run extended diagnostics to verify and diagnose the problem. Check the drive user's manual for device specific instructions on running extended diagnostic tests."
Loader Hardware A	"Loader mechanism is having trouble communicating with the drive."	"6" = "Critical"	"The changer mechanism is having difficulty communicating with the drive:"	"1. Turn the autoloader off then on." "2. Restart the operation." "3. If a problem persists, call the drive supplier help line."
Loader Stray Media	"Stray media left in loader after previous error recovery."	"6" = "Critical"	"A media has been left in the autoloader by a previous hardware fault:"	"1. Insert an empty magazine to clear the fault." "2. If the fault does not clear, turn the autoloader off and then on again." "3. If the problem persists, call the drive supplier help line."
Loader Hardware B	"Loader mechanism has a hardware fault."	"3"= "Degrading/ Warning"	"There is a problem with the autoloader mechanism."	

Table 35 - LibraryAlert AlertIndication Properties (Continued)

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Loader Door	"Changer door open."	"6" = "Critical"	"The operation has failed because the autoloader door is open:"	"1. Clear any obstructions from the autoloader door." "2. Eject the magazine and then insert it again." "3. If the fault does not clear, turn the autoloader off and then on again." "4. If the problem persists, call the drive supplier help line."
Loader Hardware C	"The loader mechanism has a hardware fault that is not mechanically related."	"6" = "Critical"	"The autoloader has a hardware fault:"	"1. Turn the autoloader off and then on again." "2. Restart the operation." "3. If the problem persists, call the drive supplier help line. Check the autoloader user's manual for device specific instructions on turning the device power on and off."
Loader Magazine	"Loader magazine not present."	"6" = "Critical"	"The autoloader cannot operate without the magazine:"	"1. Insert the magazine into the autoloader." "2. Restart the operation."
Loader Predictive Failure	"Predictive failure of loader mechanism hardware"	"3" = "Degrading/ Warning"		"A hardware failure of the changer mechanism is predicted. Call the drive supplier help line."
Load Statistics	"Drive or library powered down with media loaded."	"3" = "Degrading/ Warning"	"Media statistics have been lost at some time in the past."	

Table 35 - LibraryAlert AlertIndication Properties (Continued)

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Media Directory Invalid at Unload	"Error preventing the media directory being updated on unload."	"3" = "Degrading/ Warning"	"The directory on the media just unloaded has been corrupted."	"The directory can be rebuilt by reading all the data."
Media System area Write Failure	"Write errors while writing the system area on unload."	"6" = "Critical"	"The media just unloaded could not write its system area successfully: "	"1. Copy data to another cartridge." "2. Discard the old cartridge."
Media System Area Read Failure	"Read errors while reading the system area on load."	"6" = "Critical"	"The media system area could not be read successfully at load time: "	"Copy data to another cartridge."
No Start of Data	"Media damaged, bulk erased, or incorrect format."	"6" = "Critical"	"The start of data could not be found on the media:"	"1. Check that you are using the correct format media." "2. Discard the media or return the media to your supplier."
Loading Failure	"The drive is unable to load the media"	"6" = "Critical"	"The operation has failed because the media cannot be loaded and threaded."	"1. Remove the cartridge, inspect it as specified in the product manual, and retry the operation." "2. If the problem persists, call the drive supplier help line."
Library Hardware A	"Changer mechanism is having trouble communicating with the internal drive"	"6" = "Critical"	"The library mechanism is having difficulty communicating with the drive: "	"1. Turn the library off then on." "2. Restart the operation." "3. If the problem persists, call the library supplier help line."
Library Hardware B	"Changer mechanism has a hardware fault"	"3" = "Degrading/ Warning"		"There is a problem with the library mechanism. If problem persists, call the library supplier help line."

Table 35 - LibraryAlert AlertIndication Properties (Continued)

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Library Hardware C	"The changer mechanism has a hardware fault that requires a reset to recover."	"6" = "Critical"	"The library has a hardware fault"	"1. Reset the library." "2. Restart the operation. Check the library user's manual for device specific instructions on resetting the device."
Library Hardware D	"The changer mechanism has a hardware fault that is not mechanically related or requires a power cycle to recover."	"6" = "Critical"	"The library has a hardware fault:"	"1. Turn the library off then on again." "2. Restart the operation." "3. If the problem persists, call the library supplier help line. Check the library user's manual for device specific instructions on turning the device power on and off."
Library Diagnostic Required	"The changer mechanism may have a hardware fault which would be identified by extended diagnostics."	"3" = "Degrading/ Warning"	"The library mechanism may have a hardware fault."	Run extended diagnostics to verify and diagnose the problem. Check the library user's manual for device specific instructions on running extended diagnostic tests."
Library Interface	"The library has identified an interface fault"	"6" = "Critical"	"Bad cable"	"1. Check the cables and connections." "2. Restart the operation."
Failure Prediction	"Predictive failure of library hardware"	"3" = "Degrading/ Warning"		"A hardware failure of the library is predicted. Call the library supplier help line."

Table 35 - LibraryAlert AlertIndication Properties (Continued)

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Library Maintenance	"Library preventative maintenance required."	"3" = "Degrading/ Warning"		"Preventive maintenance of the library is required. Check the library user's manual for device specific preventative maintenance tasks, or call your library supplier help line."
Library Humidity Limits	"Library humidity limits exceeded"	"6" = "Critical"	"Library humidity range is outside the operational conditions"	
Library Temperature Limits	"Library temperature limits exceeded"	"6" = "Critical"	"Library temperature is outside the operational conditions"	
Library Voltage Limits	"Library voltage limits exceeded"	"6" = "Critical"	"Potential problem with a power supply."	
Library Stray Media	"Stray cartridge left in library after previous error recovery"	"6" = "Critical"	"Cartridge left in picker or drive"	"1. Insert an empty magazine to clear the fault." "2. If the fault does not clear, turn the library off and then on again." "3. If the problem persists, call the library supplier help line."
Library Pick Retry	"Operation to pick a cartridge from a slot had to perform an excessive number of retries before succeeding"	"3" = "Degrading/ Warning"	"There is a potential problem with the drive ejecting cartridges or with the library mechanism picking a cartridge from a slot."	"1.Run diagnostics to determine the health of the Library." "2. If the problem persists, call the library supplier help line."

Table 35 - LibraryAlert AlertIndication Properties (Continued)

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Library Place Retry	"Operation to place a cartridge in a slot had to perform an excessive number of retries before succeeding"	"3" = "Degrading/ Warning"	"Worn cartridge or bad storage slot/ magazine"	"1. No action needs to be taken at this time." "2. If the problem persists, call the library supplier help line."
Library Load Retry	"Operation to load a cartridge in a drive had to perform an excessive number of retries before succeeding"	"3" = "Degrading/ Warning"	"Worn cartridge or picker"	"Run diagnostics to determine the health of the library."
Library Door	"Library door open is preventing the library from functioning"	"6" = "Critical"	"The library has failed because the door is open:"	"1. Clear any obstructions from the library door." "2. Close the library door." "3. If the problem persists, call the library supplier help line."
Library Mailslot	"Mechanical problem with import/export mailslot"	"6" = "Critical"	"There is a mechanical problem with the library media mailslot."	"Check for wedged storage media in import/export mailslot"
Library Magazine	"Library magazine not present"	"6" = "Critical"	"Administrator has removed the library's magazine"	"1. Insert the magazine into the library." "2. Restart the operation."
Library Security	"Library door opened then closed during operation"	"3" = "Degrading/ Warning"	"Administrator is trying to remove or insert a storage media"	

Table 35 - LibraryAlert AlertIndication Properties (Continued)

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Library Security Mode	"Library security mode changed"	"2" = "Information"	"Administrator changed security mode"	"The library security mode has been changed. The library has either been put into secure mode, or the library has exited the secure mode. This is for information purposes only. No action is required."
Library Offline	"Library manually turned offline"	"2" = "Information"	"The library has been manually turned offline and is unavailable for use."	
Library Drive Offline	"Library turned internal drive offline."	"2" = "Information"	"Drive failure"	"A drive inside the library has been taken offline. This is for information purposes only. No action is required."
Library Scan Retry	"Operation to scan the bar code on a cartridge had to perform an excessive number of retries before succeeding"	"3" = "Degrading/ Warning"	"There is a potential problem with the bar code label or the scanner hardware in the library mechanism."	"1. No action needs to be taken at this time." "2. If the problem persists, call the library supplier help line."
Library Inventory	"Inconsistent media inventory"	"6" = "Critical"	"Media label has changed or bad Bar code scanner subsystem problem."	"1. Redo the library inventory to correct inconsistency." "2. Restart the operation. Check the applications user's manual or the hardware user's manual for specific instructions on redoing the library inventory."
Library Illegal Operation	"Illegal operation detected"	"3" = "Degrading/ Warning"	"A library operation has been attempted that is invalid at this time."	

Table 35 - LibraryAlert AlertIndication Properties (Continued)

Event/Alert Summary	AlertIndication "Mapped" Properties from SSC-2 and SMC-2 Specs			
	OtherAlert Type	Perceived Severity	ProbableCause Description	Recommended Action[]
	string	Uint16	string	string
Dual-Port Interface Error	"Failure of one interface port in a dual-port configuration"	"3" = "Degrading/ Warning"	"A redundant interface port on the library has failed."	
Cooling Fan Failure	"One or more fans inside the library have failed. Internal flag state only cleared when all flags are working again"	"3" = "Degrading/ Warning"	"Bad cooling Fan"	
Power Supply	"Redundant power supply failure inside the library subsystem"	"3" = "Degrading/ Warning"	"Bad Power Supply"	"A redundant power supply has failed inside the library. Check the library user's manual for instructions on replacing the failed power supply. "
Power Consumption	"Power consumption of one or more devices inside the library is outside the specified range"	"3" = "Degrading/ Warning"	"The library power consumption is outside the specified range."	
Pass Through Mechanism Failure	"Error occurred in pass-through mechanism during self test or while attempting to transfer a cartridge between library modules"	"6" = "Critical"	"A failure has occurred in the cartridge pass-through mechanism between two library modules."	
Cartridge in Pass-through Mechanism	"Cartridge left in the pass-through mechanism between two library modules"	"6" = "Critical"		"A cartridge has been left in the pass-through mechanism from a previous hardware fault. Check the library users guide for instructions on clearing this fault."
Unreadable barcode Labels	"Unable to read a bar code label on a cartridge during library inventory/ scan"	"2" = "Information"	"Bad Bar Code Labels or Scanner"	"The library was unable to read the bar code on a cartridge."

8.7 Registered Name and Version

SML_Events version 1.1.0

8.8 CIM Elements

Table 36 describes the CIM elements for SML_Events.

Table 36 - CIM Elements for SML_Events

Element Name	Requirement	Description
8.8.1 CIM_AlertIndication	Mandatory	

8.8.1 CIM_AlertIndication

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

Table 37 describes class CIM_AlertIndication.

Table 37 - SMI Referenced Properties/Methods for CIM_AlertIndication

Properties	Flags	Requirement	Description & Notes
Description		Mandatory	"LibraryAlertIndication".
AlertType		Mandatory	5 = "Device Alert".
ProbableCause		Mandatory	1 = "other".
Trending		Mandatory	1 = "Not Applicable".
SystemCreationClassName		Mandatory	CIM_ComputerSystem.
OtherSeverity		Mandatory	Specified by vendor.
EventID		Mandatory	Specified by vendor.
ProviderName		Mandatory	Specified by vendor.
SystemName		Mandatory	
AlertingManagedElement		Mandatory	
OtherAlertType		Mandatory	
PerceivedSeverity		Mandatory	
ProbableCauseDescription		Mandatory	

EXPERIMENTAL

STABLE

Clause 9: Limited Access Port Elements Subprofile

9.1 Description

Most libraries contain Limited Access Ports elements (a.k.a., mailslots, cartridge access ports, or import/export elements). This subprofile defines the classes necessary to publish information about these common components.

9.1.1 Instance Diagram

Figure 13 and Figure 14 show the relationship between LimitedAccessPorts and other portions of the Storage Library profile.

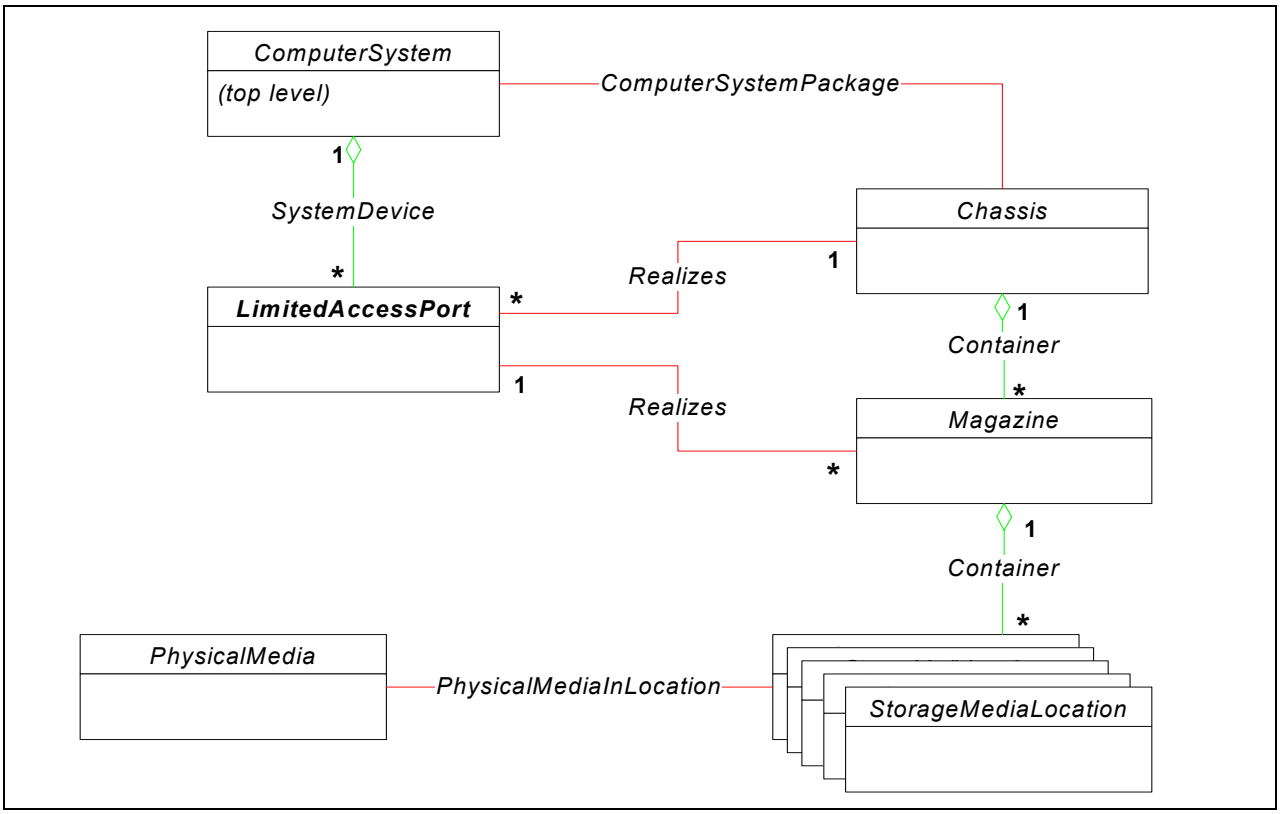

Figure 13 - Tape Libraries with Magazines in LimitedAccessPorts

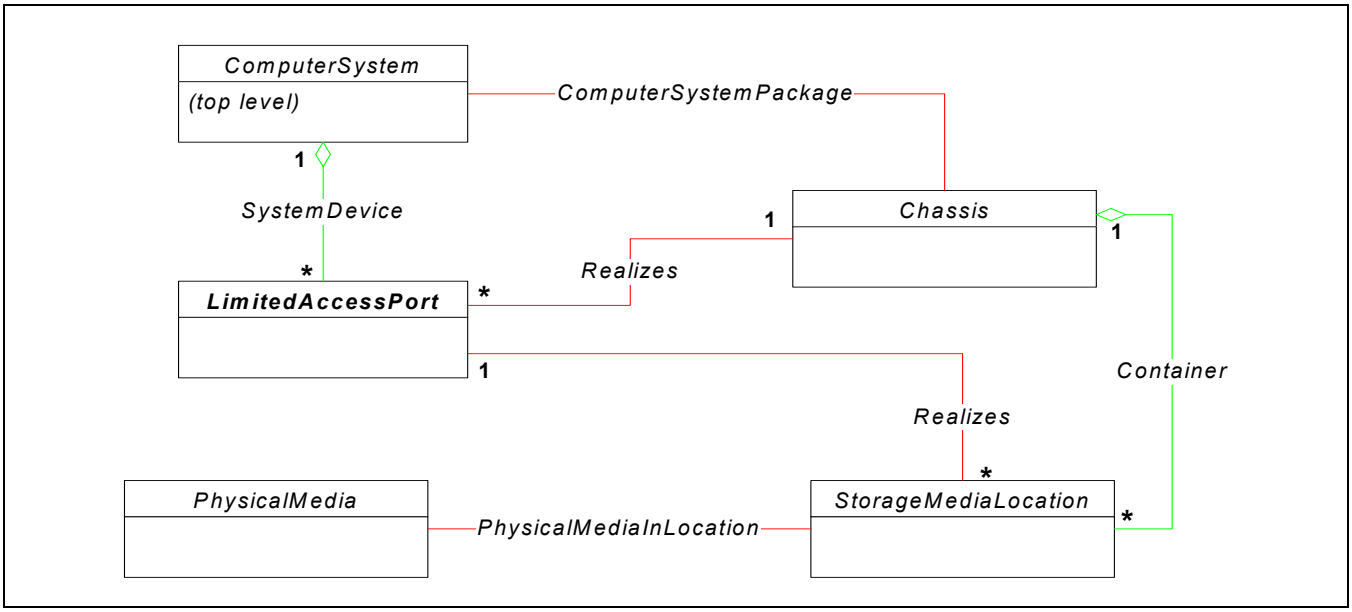

Figure 14 - Tape Libraries with no Magazines in LimitedAccessPorts

9.2 Health and Fault Management Considerations

Not defined in this standard.

9.3 Cascading Considerations

Not defined in this standard.

9.4 Supported Subprofiles and Packages

None.

9.5 Methods of the Profile

None.

9.5.1 Client Considerations and Recipes

None

9.6 Registered Name and Version

Storage Library Limited Access Port Elements version 1.2.0

9.7 CIM Elements

Table 38 describes the CIM elements for Storage Library Limited Access Port Elements.

Table 38 - CIM Elements for Storage Library Limited Access Port Elements

Element Name	Requirement	Description
9.7.1 CIM_Container	Mandatory	The containment relationship of Magazines within a Chassis or StorageMediaLocations within a Magazine.
9.7.2 CIM_LimitedAccessPort	Mandatory	LimitedAccessPorts represent hardware that transports physical media into or out of a Storage Library. They are identified as 'limited' since these ports do not provide access to ALL the PhysicalMedia or StorageMediaLocations in a Library, but only to a subset.
9.7.3 CIM_Magazine	Mandatory	
9.7.4 CIM_Realizes	Mandatory	The relationship between a LimitedAccessPort and the StorageMediaLocations, Magazines or Chassis to which it has access.
9.7.5 CIM_SystemDevice	Mandatory	The relationship between a LimitedAccessPort and its hosting top-level ComputerSystem which represents the Storage Library.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_LimitedAccessPort	Mandatory	Creation of an instance of LimitedAccessPort.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_LimitedAccessPort	Mandatory	Deletion of an instance of LimitedAccessPort.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_LimitedAccessPort AND SourceInstance.OperationalStatus <> PreviousInstance.OperationalStatus	Mandatory	Deprecated WQL -Change in OperationalStatus of a LimitedAccessPort.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_LimitedAccessPort AND SourceInstance.CIM_LimitedAccessPort::OperationalStatus <> PreviousInstance.CIM_LimitedAccessPort::OperationalStatus	Mandatory	CQL -Change in OperationalStatus of a LimitedAccessPort.

9.7.1 CIM_Container

Created By: Static

Modified By: Static

Deleted By: Static
Requirement: Mandatory

Table 39 describes class CIM_Container.

Table 39 - SMI Referenced Properties/Methods for CIM_Container

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

9.7.2 CIM_LimitedAccessPort

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

Table 40 describes class CIM_LimitedAccessPort.

Table 40 - SMI Referenced Properties/Methods for CIM_LimitedAccessPort

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
DeviceID		Mandatory	
Extended		Mandatory	When true, the port's StorageMediaLocations are accessible to a human operator. When false, the StorageMediaLocations are accessible to a PickerElement.
ElementName		Mandatory	User-friendly name.
OperationalStatus		Mandatory	Status of the LimitedAccessPort.
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.

9.7.3 CIM_Magazine

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

Table 41 describes class CIM_Magazine.

Table 41 - SMI Referenced Properties/Methods for CIM_Magazine

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Tag		Mandatory	
LocationType		Mandatory	"Magazine".
LocationCoordinates		Mandatory	
MediaTypesSupported		Mandatory	
MediaCapacity		Mandatory	The maximum number of PhysicalMedia that this StorageMediaLocation can hold.
PhysicalLabels		Optional	
LabelStates		Optional	
LabelFormats		Optional	

9.7.4 CIM_Realizes

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

Table 42 describes class CIM_Realizes.

Table 42 - SMI Referenced Properties/Methods for CIM_Realizes

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

9.7.5 CIM_SystemDevice

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

Table 43 describes class CIM_SystemDevice.

Table 43 - SMI Referenced Properties/Methods for CIM_SystemDevice

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

STABLE

EXPERIMENTAL

Clause 10: Media Movement Subprofile

10.1 Description

The Media Movement Subprofile defines a method to physically move a PhysicalMedia element from its current StorageMediaLocation to another StorageMediaLocation within the library with which the media is compatible. Such a method is convenient for purposes including library maintenance, self test, and demonstration. The method is implemented by a HostedService associated with the ComputerSystem which models the storage library. The method supports asynchronous operation according to the Job Control Subprofile.

Figure 15 illustrates the subprofile from the library perspective.

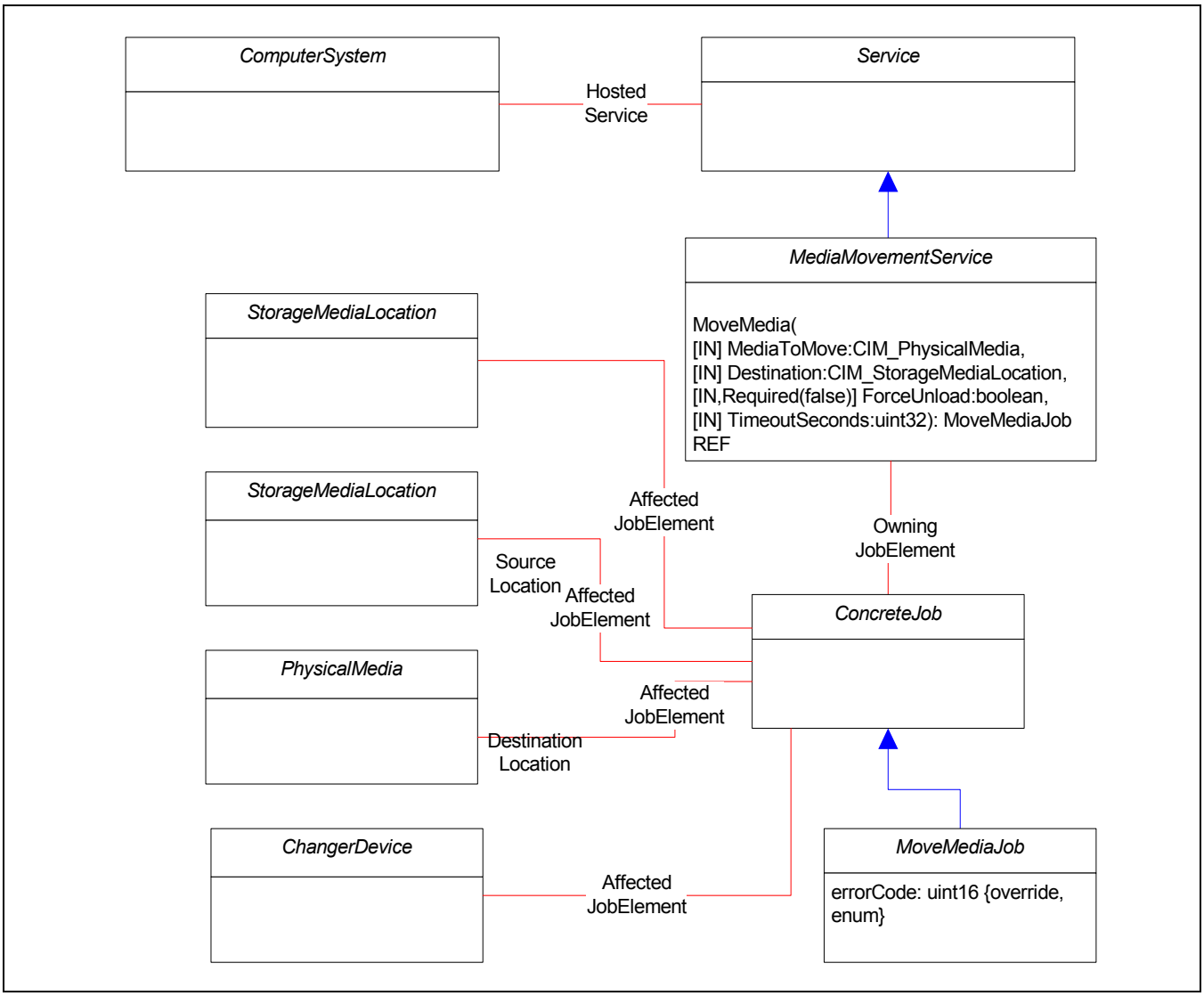

Figure 15 - Storage Library Centric View

When the move media operation is performed, the storage library shall physically move the medium, and then update the storage library's CIM object model. In particular, the StorageMediaInLocation association between the

PhyscialMedia instance and the source StorageMediaLocation instance shall be removed and a new association made between the PhysicalMedia instance and the destination StorageMediaLocation. This is illustrated in Figure 16.

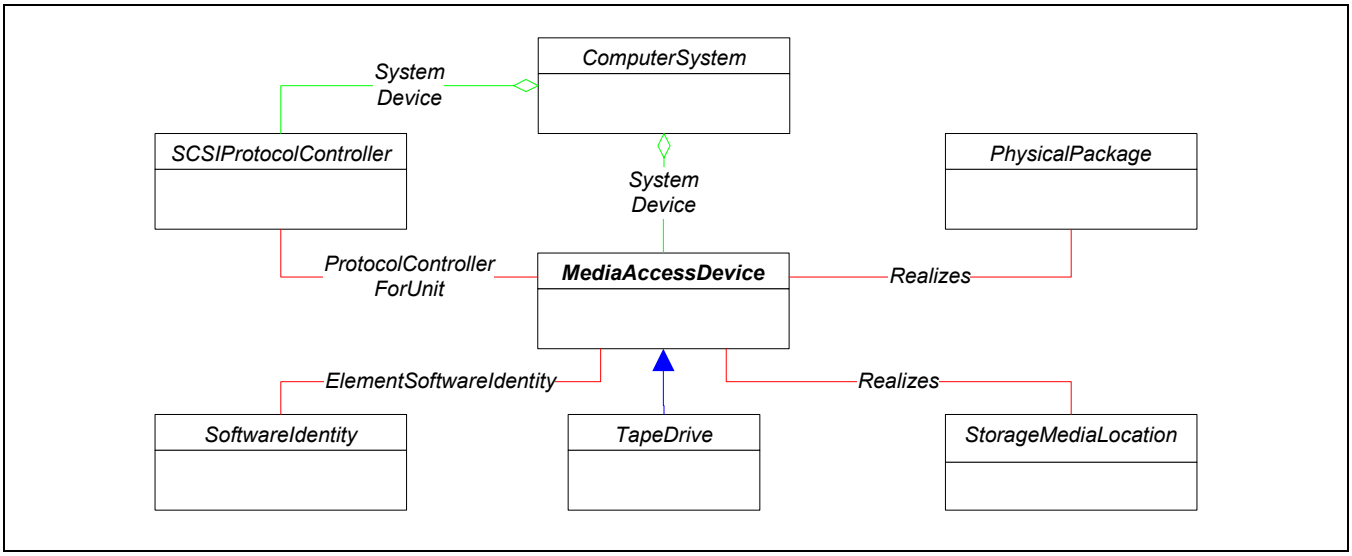

Figure 16 - Media-centrc View

10.2 Health and Fault Management Considerations

10.2.1 NULL Instance Handling

If a non-null instance of ConcreteJob is returned by the MoveMedia method, the implementation shall report errors which occur during the execution of the job through the ConcreteJob.GetError() method. See *Storage Management Technical Specification, Part 2 Common Profiles, 1.4.0 Rev 6* Clause 10: Media Movement Subprofile for details.

10.2.2 8.1 Media Movement Subprofile Standard Messages

The standard messages specific to this profile are listed in Table 44.

Table 44 - Media Movement Standard Messages

Message ID	Message Name
1	Source Media not Found
2	Destination Location Full
3	Invalid Source Media
4	Invalid Destination Location
5	Media not Compatible with Destination
6	Reservation Conflict
7	Busy
8	Hardware Error

Table 44 - Media Movement Standard Messages (Continued)

Message ID	Message Name
9	Internal Model Error
10	Command Sequence Error

10.3 Cascading Considerations

Not defined in this standard.

10.4 Supported Subprofiles and Packages

None.

10.5 Methods of the Profile

10.5.1 Moving a piece of PhysicalMedia

```
uint32 MoveMedia(
    [OUT, Description("Reference to the job (may be null if job completed.)")]
     CIM_ConcreteJob REF MoveMediaJob,
   [IN, Description( "The piece of media to be moved" ) ]
   CIM_PhysicalMedia REF MediaToMove,
   [IN, Description( "The destination location" ) ]
   CIM_StorageMediaLocation REF Destination,
     [IN, Required(false),
        Description( "Optional parameter instructing the storage library to "
         "first unload the media if it is loaded in a MediaAccessDevice." ) ]
     boolean ForceUnload,
  [IN, Required(false),
       Description( "The timeout time in seconds" ) ]
  unit32 Timeout )
```

Error returns are:

```
{ "Job Completed with No Error", "Not Supported", "Unknown", "Timeout",
  "Failed", "Invalid Parameter", "In Use", "DMTF Reserved",
  "Method Parameters Checked - Job Started", "Busy", "Method Reserved",
  "Vendor Specific" }
```

The MoveMedia method takes as input references to the media to be moved, the destination location, and a timeout value. The method attempts to initiate a process on the Storage Library which will perform the media movement. If the process is successfully initiated, the MoveMedia returns a ConcreteJob object and an integer return code indicating the status of the job creation. If a non-null instance of ConcreteJob is returned, the instance shall be associated with an instance of MethodResult as specified by the Job Control Subprofile. See *Storage Management Technical Specification, Part 2 Common Profiles, 1.4.0 Rev 6* Clause 25: Job Control Subprofile for details of job creation and execution.

10.5.1.1 Timeout parameter

The optional Timeout parameter allows the MediaMovementService process or a sub-process to handle job timeout rather than delegating the responsibility to the SMI client. If the Timeout parameter is omitted (set to “null”), the method shall use the library’s default behavior, which may be vendor or library specific.

10.5.1.2 ForceUnload parameter

When set to “true”, the optional ForceUnload parameter instructs the Storage Library to first unload the PhysicalMedia if it is loaded in a MediaAccessDevice. If the ForceUnload parameter is set to “false” and the PhysicalMedia is loaded in a MediaAccessDevice, the job shall fail and the ConcreteJob’s GetError() method shall return an instance of

Error indicating “Media Loaded in Access Device”, an error message specific to the Media Movement Subprofile. If the ForceUnload parameter is omitted (set to “null”), the method shall use the library’s default behavior, which may be vendor or library specific.

10.6 Client Considerations and Recipes

10.6.1 Concurrent library access by SMI clients and other applications.

The MoveMedia method introduces an alternate path to modify the configuration of the storage library, possibly interfering with the operation of other applications using the library concurrently. The MoveMedia method shall be used with caution in situations where applications other than the SMI client are moving media in the storage library.

10.6.2 Use of the ForceUnload parameter

Forcing a MediaAccessDevice to unload media while in use by other applications may cause data loss.

10.6.3 Job Lifecycle Indications

SMI Servers implementing the Job Control profile are required to support a set of indications which indicate transitions in the operational status of the job. In particular, an indication shall be provided when a job stops, either successfully or with an error condition. The server may also generate indications for change in job status or percent complete. See 25.8 "CIM Elements" in Clause 25: Job Control Subprofile of the *Storage Management Technical Specification, Part 2 Common Profiles, 1.4.0 Rev 6* for indication subscription details.

10.7 Registered Name and Version

Storage Library Media Movement version 1.1.0

10.8 CIM Elements

Table 45 describes the CIM elements for Storage Library Media Movement.

Table 45 - CIM Elements for Storage Library Media Movement

Element Name	Requirement	Description
10.8.1 CIM_HostedService	Mandatory	The relationship between the top-level ComputerSystem representing the Storage Library and the MediaMovementService.
10.8.2 SNIA_MediaMovementService	Mandatory	

10.8.1 CIM_HostedService

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

Table 46 describes class CIM_HostedService.

Table 46 - SMI Referenced Properties/Methods for CIM_HostedService

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

10.8.2 SNIA_MediaMovementService

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

Table 47 describes class SNIA_MediaMovementService.

Table 47 - SMI Referenced Properties/Methods for SNIA_MediaMovementService

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
Name		Mandatory	
MoveMedia()		Mandatory	

EXPERIMENTAL

EXPERIMENTAL

Clause 11: Partitioned Tape Library Profile

11.1 Description

11.1.1 Overview

This profile describes the model for a Partitioned Tape Library (PTL). Partitioning allows an organization to share a physical tape library across multiple clients with disparate departmental needs. Using a single physical infrastructure, it permits individual departments to preserve their own operating environment and security policies. For instance, instead of buying three libraries with 100 slots each and individual service agreements, a single 300-slot library can be purchased cutting down significantly on the total cost of ownership (TCO).

11.1.2 PTL Model

Figure 17 illustrates the major components of the PTL model. The Partitioned Tape Library System component is the central component that is responsible for configuring and managing all the partitions of the library. The ComputerSystem class instance contains the Dedicated property value of “Partitioned Library System” and acts as the top-level ComputerSystem instance for the rest of the components.

Each partition of the library is represented by the left-most column on elements with its ComputerSystem Dedicated property containing a value of “Partition”. These elements are modeled along the same lines as the physical Storage Library profile.

The middle set of elements represent the unallocated set of resources (tape drives, physical tapes, slots, changer, etc.) and modeled as a partition with the ComputerSystem Dedicated property containing a value of “Unallocated Partition”.

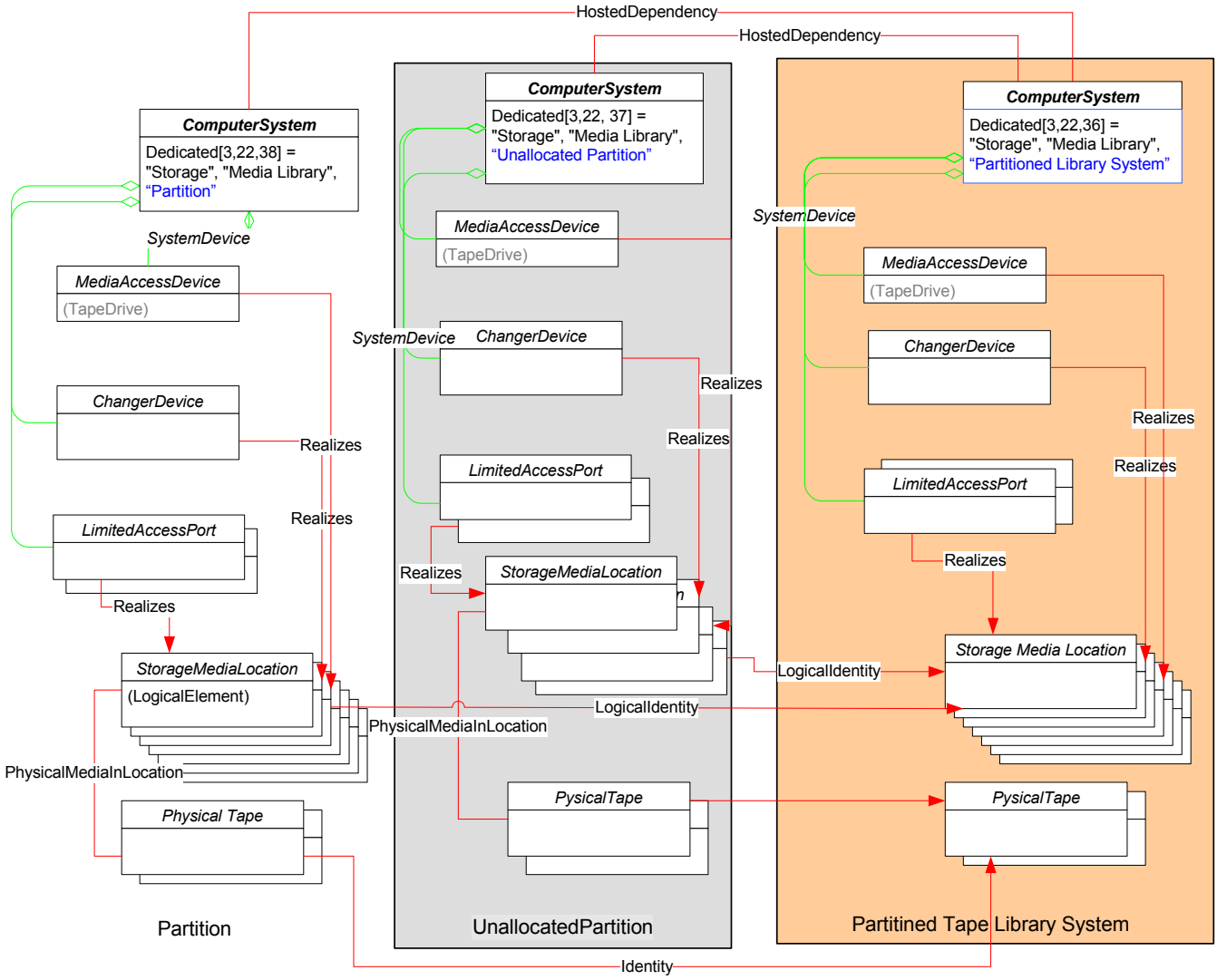

Figure 17 - Partitioned Tape Library System Model

11.1.3 PTL Configuration

Figure 18 shows the model related to the management of partitions in a PTL system.

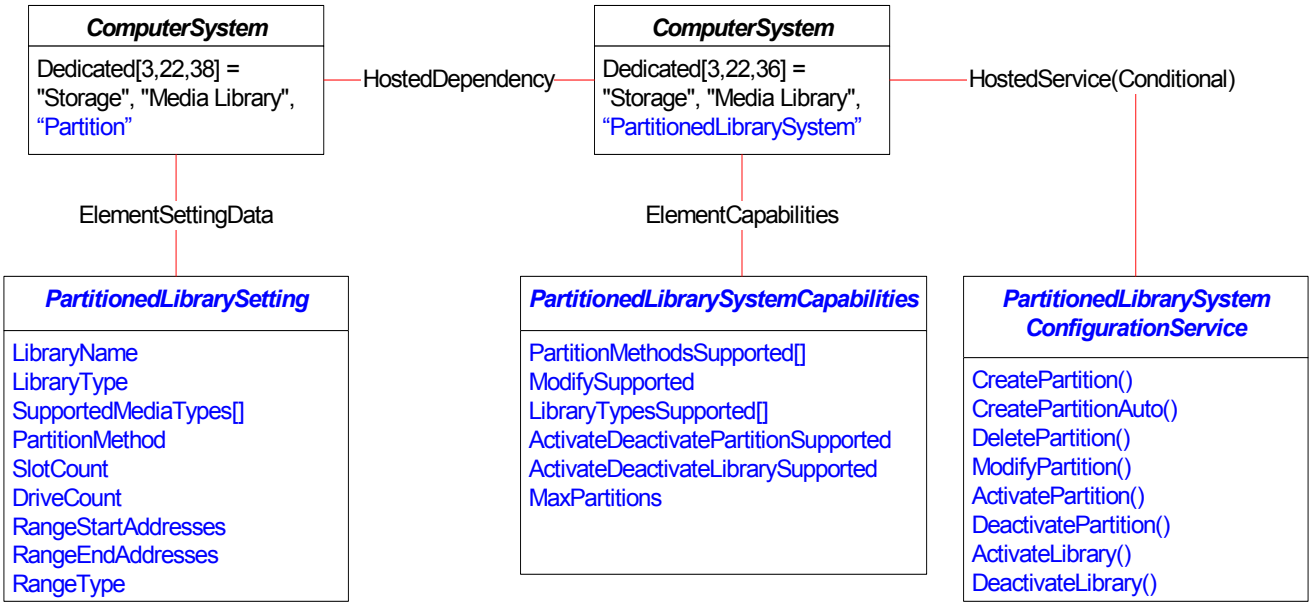

Figure 18 - Partitioned Tape Library Configuration Model

The PartitionedLibrarySystemCapabilities class contains properties that define the capabilities of the PTL system where as the PartitionedLibrarySystemConfigurationService class defines methods to create and manage the partitions in the PTL system.

Depending on the PTL system, there are variations on how you create a partition as listed in the PartitionMethodsSupported property in the capabilities class. The value map for this property defines the possible enumerations.

In **Auto** partitioning, given number of partitions N, the PTL system can create N partitions. How the library chooses to divide the available resources depends on the specific library. One way is to divide the resources equally.

In **Simple** partitioning, a partition is created by providing the number of slots and drives the partition should contain.

In **Slot** partitioning, a set of ranges of Slots (Storage Media Locations) are specified for the partition.

In **VolSer** partitioning, a partition can be created by assigning a set of ranges of cartridge serial numbers that belong to the partition. You also assign drives to the partition by specifying the storage media locations for the drives.

The parameters for all the above variations of the creation method are specified using the PartitionedLibrarySetting class instance.

A PTL system may not support any of the create methods at all in a monitor-only implementation.

11.1.4 PTL Configuration Methods

CreatePartition(PartitionSetting) creates a single partition using one of the variations defined above except for Auto partitioning.

PartitionSetting is a reference to a PartitionedLibrarySetting instance that contain the create parameter values including the PartitionMethod to be used.

CreatePartitionAuto(NumberOfPartitions) creates a number of partitions as given. How the resources are divided among the partitions is implementation-dependent.

ModifyPartition(PartitionSetting) takes a new setting instance as target and modifies the partition definition.

DeletePartition(ThePartition) deletes the referenced partition.

ActivatePartition(ThePartition) Once a partition is created or modified, the referenced partition can be brought online using this method.

DeactivatePartition(ThePartition) Before you delete or modify a partition, you need to bring the partition offline by using this method.

ActivateLibrary() In some libraries, you first create all the partitions and then activate all the partitions at once. This method activates all the partitions. The ActivateDeactivateLibrarySupported property in the capabilities class defines if a particular library has this requirement.

DeactivateLibrary() Similar to the activate, this method deactivates all the partitions so that some or all the partitions can be modified or deleted.

11.2 Health and Fault Management Consideration

Not defined in this standard.

11.3 Cascading Considerations

Not defined in this standard.

11.4 Supported Profiles, Subprofiles, and Packages

Table 48 describes the supported profiles for Partitioned Tape Library.

Table 48 - Supported Profiles for Partitioned Tape Library

Profile Name	Organization	Version	Requirement	Description
Access Points	SNIA	1.3.0	Optional	
FC Target Ports	SNIA	1.4.0	Optional	
SAS Target Ports	SNIA	1.4.0	Optional	
SPI Target Ports	SNIA	1.4.0	Optional	
Health	SNIA	1.2.0	Mandatory	
Software	SNIA	1.4.0	Optional	
Storage Library	SNIA	1.2.0	Optional	
Indication	SNIA	1.4.0	Mandatory	
Multiple Computer System	SNIA	1.2.0	Optional	
Masking and Mapping	SNIA	1.4.0	Optional	
Storage Library Element Counting	SNIA	1.1.0	Optional	

Table 48 - Supported Profiles for Partitioned Tape Library

Profile Name	Organization	Version	Requirement	Description
Storage Library Capacity	SNIA	1.1.0	Optional	
Storage Library Limited Access Port Elements	SNIA	1.2.0	Optional	
Storage Library Media Movement	SNIA	1.1.0	Optional	
Location	SNIA	1.4.0	Optional	

11.5 Client Considerations and Recipes

Not defined in this profile.

11.6 Registered Name and Version

Partitioned Tape Library version 1.4.0

11.7 CIM Elements

Table 49 describes the CIM elements for Partitioned Tape Library.

Table 49 - CIM Elements for Partitioned Tape Library

Element Name	Requirement	Description
11.7.1 CIM_ChangerDevice	Optional	The media changer for a PTL system (this is the physical pool).
11.7.2 CIM_Chassis (PTL System)	Optional	The box for a PTL.
11.7.3 CIM_ComputerSystemPackage (PTL System to Chassis)	Mandatory	This association links Chassis to the scoping system.
11.7.4 CIM_ConcreteIdentity (Slots to Slots)	Mandatory	This association links ports to the slots.
11.7.5 CIM_Container (Chassis to slots)	Mandatory	This association links Slots to the chassis.
11.7.6 CIM_ElementCapabilities	Optional	
11.7.7 CIM_ElementSettingData	Optional	
11.7.8 CIM_HostedDependency (PTLSystem to Partition)	Mandatory	This association links the PTLSystem ComputerSystem object to the Partition ComputerSystem objects including the unallocated; hence 1ormore.

Table 49 - CIM Elements for Partitioned Tape Library

Element Name	Requirement	Description
11.7.9 CIM_HostedDependency (PTLSystem to Unallocated Partition)	Mandatory	This association links the PTLSystem ComputerSystem object to the Unnallocated Partition.
11.7.10 CIM_LimitedAccessPort	Optional	The media export port for a PTL system (this is the physical pool).
11.7.11 CIM_MediaAccessDevice	Optional	If unknown, set to False.
11.7.12 CIM_PhysicalMediaInLocation	Optional	This association links media to the slots.
11.7.13 CIM_PhysicalTape	Mandatory	The media in the PTL Collection.
11.7.14 CIM_Product	Optional	Asset information for the system.
11.7.15 CIM_ProductElementComponent (PTL System)	Optional	
11.7.16 CIM_Realizes (Slots to Changers)	Mandatory	This association links changers to the slots.
11.7.17 CIM_Realizes (Slots to Ports)	Mandatory	This association links ports to the slots.
11.7.18 CIM_Realizes (Slots to TapeDrive)	Mandatory	This association links drives to the slots.
11.7.19 CIM_StorageMediaLocation	Optional	The slots and drive slots in a partition tape library system (the physical pool).
11.7.20 CIM_SystemDevice (PTL System to ChangerDevice)	Mandatory	This association links ChangerDevice to the scoping system.
11.7.21 CIM_SystemDevice (PTL System to LimitedAccessPort)	Mandatory	This association links LimitedAccessDevice to the scoping system.
11.7.22 CIM_SystemDevice (PTL System to MediaAccessDevice)	Mandatory	This association links MediaAccessDevice to the scoping system.
11.7.23 SNIA_ComputerSystem (PTL System)	Mandatory	'Top level' system that represents the entire Virtual Tape Library.
11.7.24 SNIA_ComputerSystem (Partition)	Optional	'Top level' system that represents a Partition within a Tape Library.
11.7.25 SNIA_ComputerSystem (Unallocated Partition)	Mandatory	'Top level' system that represents the unallocated portion of the physical library .
11.7.26 SNIA_PartitionedLibrarySetting	Optional	Settings used to create the PTL.
11.7.27 SNIA_PartitionedLibrarySystemCapabilities	Optional	Features supported in PTL Service.
11.7.28 SNIA_PartitionedLibrarySystemConfiguration Service	Optional	Services used to set up the PTLS hardware.

Table 49 - CIM Elements for Partitioned Tape Library

Element Name	Requirement	Description
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA SNIA_ComputerSystem AND ANY SourceInstance.SNIA_ComputerSystem::Dedicated[*] = 38	Mandatory	CQL -Partition was created.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_ProtocolControllerForUnit	Mandatory	
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA SNIA_ComputerSystem AND ANY SourceInstance.SNIA_ComputerSystem::Dedicated[*] = 38	Mandatory	CQL -Partition was deleted.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_ProtocolControllerForUnit	Mandatory	
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA SNIA_ComputerSystem AND ANY SourceInstance.SNIA_ComputerSystem::Dedicated[*] = 38 AND SourceIsntance.SNIA_ComputerSystem::OperationalStatus <> PreviousInstance.SNIA_ComputerSystem::OperationalStatus	Mandatory	CQL -Status of a Partition or a PTL System has changed.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_PhysicalTape	Mandatory	

11.7.1 CIM_ChangerDevice

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic
Requirement: Optional

Table 50 describes class CIM_ChangerDevice.

Table 50 - SMI Referenced Properties/Methods for CIM_ChangerDevice

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	

Table 50 - SMI Referenced Properties/Methods for CIM_ChangerDevice

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
DeviceID		Mandatory	

11.7.2 CIM_Chassis (PTL System)

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic
Requirement: Optional

Table 51 describes class CIM_Chassis (PTL System).

Table 51 - SMI Referenced Properties/Methods for CIM_Chassis (PTL System)

Properties	Flags	Requirement	Description & Notes
Tag		Mandatory	
CreationClassName		Mandatory	
PackageType		Mandatory	Shall be 3 (ChassisFrame).
ChassisPackageType		Mandatory	
Manufacturer		Optional	
Model		Optional	
SerialNumber		Optional	
PartNumber		Optional	
SKU		Optional	
VendorCompatibilityStrings		Optional	
ElementName		Optional	

11.7.3 CIM_ComputerSystemPackage (PTL System to Chassis)

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

Table 52 describes class CIM_ComputerSystemPackage (PTL System to Chassis).

Table 52 - SMI Referenced Properties/Methods for CIM_ComputerSystemPackage (PTL System to Chassis)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

11.7.4 CIM_ConcreteIdentity (Slots to Slots)

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

Table 53 describes class CIM_ConcreteIdentity (Slots to Slots).

Table 53 - SMI Referenced Properties/Methods for CIM_ConcreteIdentity (Slots to Slots)

Properties	Flags	Requirement	Description & Notes
SystemElement		Mandatory	
SameElement		Mandatory	

11.7.5 CIM_Container (Chassis to slots)

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

Table 54 describes class CIM_Container (Chassis to slots).

Table 54 - SMI Referenced Properties/Methods for CIM_Container (Chassis to slots)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

11.7.6 CIM_ElementCapabilities

Created By: Static
Modified By: Static

Deleted By: Static
Requirement: Optional

Table 55 describes class CIM_ElementCapabilities.

Table 55 - SMI Referenced Properties/Methods for CIM_ElementCapabilities

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	
Capabilities		Mandatory	

11.7.7 CIM_ElementSettingData

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

Table 56 describes class CIM_ElementSettingData.

Table 56 - SMI Referenced Properties/Methods for CIM_ElementSettingData

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	
SettingData		Mandatory	

11.7.8 CIM_HostedDependency (PTLSystem to Partition)

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

Table 57 describes class CIM_HostedDependency (PTLSystem to Partition).

Table 57 - SMI Referenced Properties/Methods for CIM_HostedDependency (PTLSystem to Partition)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	PTLSystem ComputerSystem object.
Dependent		Mandatory	Partition ComputerSystem object.

11.7.9 CIM_HostedDependency (PTLSystem to Unallocated Partition)

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

Table 58 describes class CIM_HostedDependency (PTLSystem to Unallocated Partition).

Table 58 - SMI Referenced Properties/Methods for CIM_HostedDependency (PTLSystem to Unallocated Partition)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	PTLSystem ComputerSystem object.
Dependent		Mandatory	Partition ComputerSystem object.

11.7.10 CIM_LimitedAccessPort

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic
Requirement: Optional

Table 59 describes class CIM_LimitedAccessPort.

Table 59 - SMI Referenced Properties/Methods for CIM_LimitedAccessPort

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
DeviceID		Mandatory	
OperationalStatus		Optional	
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.

11.7.11 CIM_MediaAccessDevice

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic

Requirement: Optional

Table 60 describes class CIM_MediaAccessDevice.

Table 60 - SMI Referenced Properties/Methods for CIM_MediaAccessDevice

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
DeviceID		Mandatory	
OperationalStatus		Optional	
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.
NeedsCleaning		Mandatory	
MountCount		Optional	

11.7.12 CIM_PhysicalMediaInLocation

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

Table 61 describes class CIM_PhysicalMediaInLocation.

Table 61 - SMI Referenced Properties/Methods for CIM_PhysicalMediaInLocation

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

11.7.13 CIM_PhysicalTape

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

11.7.14 CIM_Product

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

Table 62 describes class CIM_Product.

Table 62 - SMI Referenced Properties/Methods for CIM_Product

Properties	Flags	Requirement	Description & Notes
ElementName		Mandatory	
Name		Mandatory	
IdentifyingNumber		Mandatory	
Vendor		Mandatory	
Version		Mandatory	

11.7.15 CIM_ProductElementComponent (PTL System)

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

Table 63 describes class CIM_ProductElementComponent (PTL System).

Table 63 - SMI Referenced Properties/Methods for CIM_ProductElementComponent (PTL System)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

11.7.16 CIM_Realizes (Slots to Changers)

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

Table 64 describes class CIM_Realizes (Slots to Changers).

Table 64 - SMI Referenced Properties/Methods for CIM_Realizes (Slots to Changers)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

11.7.17 CIM_Realizes (Slots to Ports)

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

Table 65 describes class CIM_Realizes (Slots to Ports).

Table 65 - SMI Referenced Properties/Methods for CIM_Realizes (Slots to Ports)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

11.7.18 CIM_Realizes (Slots to TapeDrive)

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

Table 66 describes class CIM_Realizes (Slots to TapeDrive).

Table 66 - SMI Referenced Properties/Methods for CIM_Realizes (Slots to TapeDrive)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

11.7.19 CIM_StorageMediaLocation

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic

Requirement: Optional

Table 67 describes class CIM_StorageMediaLocation.

Table 67 - SMI Referenced Properties/Methods for CIM_StorageMediaLocation

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Tag		Mandatory	
LocationType		Mandatory	Slot, MediaAccessDevice, or Limited Access Port.
LocationCoordinates		Mandatory	
MediaTypesSupported		Mandatory	
MediaCapacity		Mandatory	

11.7.20 CIM_SystemDevice (PTL System to ChangerDevice)

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

Table 68 describes class CIM_SystemDevice (PTL System to ChangerDevice).

Table 68 - SMI Referenced Properties/Methods for CIM_SystemDevice (PTL System to Changer-Device)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

11.7.21 CIM_SystemDevice (PTL System to LimitedAccessPort)

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

Table 69 describes class CIM_SystemDevice (PTL System to LimitedAccessPort).

Table 69 - SMI Referenced Properties/Methods for CIM_SystemDevice (PTL System to LimitedAccessPort)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

11.7.22 CIM_SystemDevice (PTL System to MediaAccessDevice)

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

Table 70 describes class CIM_SystemDevice (PTL System to MediaAccessDevice).

Table 70 - SMI Referenced Properties/Methods for CIM_SystemDevice (PTL System to MediaAccessDevice)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

11.7.23 SNIA_ComputerSystem (PTL System)

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

Table 71 describes class SNIA_ComputerSystem (PTL System).

Table 71 - SMI Referenced Properties/Methods for SNIA_ComputerSystem (PTL System)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Name		Mandatory	Unique identifier for the PTL System. This should take the form of a string consisting of Vendor+Product+SerialNumber, derived from SCSI Inquiry Pages.
Dedicated		Mandatory	Indicates that this computer system is dedicated to operation as a PTL system.

Table 71 - SMI Referenced Properties/Methods for SNIA_ComputerSystem (PTL System)

Properties	Flags	Requirement	Description & Notes
NameFormat		Mandatory	Format for Name property. **HID** is a required format. Others are optional.
OperationalStatus		Mandatory	Overall status of the system.
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.
ElementName		Mandatory	User friendly name.
PrimaryOwnerContact	M	Optional	Contact details for PTL system owner.
PrimaryOwnerName	M	Optional	Owner of the PTL System.
OtherIdentifyingInfo		Optional	Other data that could be used to identify the PTL system.
IdentifyingDescriptions		Optional	Provides explanations and details for the entries in the OtherIdentifyingInfo property.

11.7.24 SNIA_ComputerSystem (Partition)

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic
Requirement: Optional

Table 72 describes class SNIA_ComputerSystem (Partition).

Table 72 - SMI Referenced Properties/Methods for SNIA_ComputerSystem (Partition)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Name		Mandatory	
Dedicated		Mandatory	Indicates that this computer system is dedicated to operation as a PTL.
NameFormat		Mandatory	Format for Name property.
OperationalStatus		Mandatory	Overall status of the system.
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.
ElementName		Mandatory	User friendly name.
PrimaryOwnerContact	M	Optional	Contact details for PTL owner.
PrimaryOwnerName	M	Optional	Owner of the PTL.

Table 72 - SMI Referenced Properties/Methods for SNIA_ComputerSystem (Partition)

Properties	Flags	Requirement	Description & Notes
OtherIdentifyingInfo		Optional	Other data that could be used to identify the PTL.
IdentifyingDescriptions		Optional	Provides explanations and details for the entries in the OtherIdentifyingInfo property.

11.7.25 SNIA_ComputerSystem (Unallocated Partition)

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

Table 73 describes class SNIA_ComputerSystem (Unallocated Partition).

Table 73 - SMI Referenced Properties/Methods for SNIA_ComputerSystem (Unallocated Partition)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Name		Mandatory	Unique identifier for the PTL System. This should take the form of a string consisting of Vendor+Product+SerialNumber, derived from SCSI Inquiry Pages.
Dedicated		Mandatory	Indicates that this computer system is dedicated to operation as a PTL system.
NameFormat		Mandatory	Format for Name property. **HID** is a required format. Others are optional.
OperationalStatus		Mandatory	Overall status of the system.
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.
ElementName		Mandatory	User friendly name.
PrimaryOwnerContact	M	Optional	Contact details for PTL system owner.
PrimaryOwnerName	M	Optional	Owner of the PTL System.
OtherIdentifyingInfo		Optional	Other data that could be used to identify the PTL system.
IdentifyingDescriptions		Optional	Provides explanations and details for the entries in the OtherIdentifyingInfo property.

11.7.26 SNIA_PartitionedLibrarySetting

Created By: Static

Modified By: Static
Deleted By: Static
Requirement: Optional

Table 74 describes class SNIA_PartitionedLibrarySetting.

Table 74 - SMI Referenced Properties/Methods for SNIA_PartitionedLibrarySetting

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
LibraryType		Mandatory	
SupportedMediaTypes		Mandatory	
LibraryName		Optional	If present, this shall be the name of the Partition associated with these settings.
SlotCount		Optional	If present, this shall be the number of slots in the Partition associated with these settings.
DriveCount		Optional	If present, this shall be the number of drives in the Partition associated with these settings.
PartitionMethod		Mandatory	
RangeStartAddresses		Optional	If present, this shall be the starting addresses for ranges of StorageMediaLocations or volume serial numbers of tape cartridges that make up partition depending on creation method.
RangeEndAddresses		Optional	If present, this shall be the ending addresses for ranges of StorageMediaLocations or volume serial number of tape cartridges that make up partition depending on creation method.
RangeType		Optional	If present, this shall be the type of the range (StorageMediaLocation addresses or cartridge volume serial numbers) given in RangeStartAddresses and RangeEndAddresses.

11.7.27 SNIA_PartitionedLibrarySystemCapabilities

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

Table 75 describes class SNIA_PartitionedLibrarySystemCapabilities.

Table 75 - SMI Referenced Properties/Methods for SNIA_PartitionedLibrarySystemCapabilities

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
PartitionMethodsSupported		Mandatory	
MaxPartitions		Mandatory	
LibraryTypesSupported		Optional	
ModifySupported		Mandatory	
ActivateDeactivatePartitionSupported		Mandatory	
ActivateDeactivateLibrarySupported		Mandatory	

11.7.28 SNIA_PartitionedLibrarySystemConfigurationService

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

Table 76 describes class SNIA_PartitionedLibrarySystemConfigurationService.

Table 76 - SMI Referenced Properties/Methods for SNIA_PartitionedLibrarySystemConfigurationService

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
Name		Mandatory	
CreatePartition()		Optional	Creates a partition using one of the supported methods.
DeletePartition()		Optional	Delete an existing partition.
CreatePartitionAuto()		Optional	Creates a partition by equally distributing the resources among the specified numbers.
ModifyPartition()		Optional	Modify an existing partition using the same LibrarySetting.
ActivatePartition()		Optional	Make a partition active.

Table 76 - SMI Referenced Properties/Methods for SNIA_PartitionedLibrarySystemConfigurationService

Properties	Flags	Requirement	Description & Notes
DeactivatePartition()		Optional	Make a partition inactive.
ActivateLibrary()		Optional	Activate all partitions.
DeactivateLibrary()		Optional	Deactivate all partitions.

EXPERIMENTAL

EXPERIMENTAL

Clause 12: Virtual Tape Library Profile

12.1 Description

12.1.1 Overview

This profile describes the model for a Virtual Library System. The Virtual Library System uses disk and/or tape storage to emulate one or more tape libraries. A Virtual Library System can use local storage (arrays, JBOD, or tape libraries) or connect to external storage. In the case of local storage the Virtual Library System model may optionally include Storage Media Library as a supported profile. Figure 19 shows the basic components of the Virtual Library System.

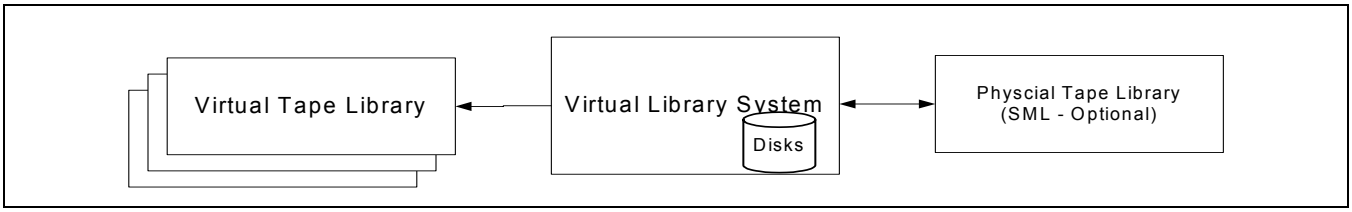

Figure 19 - Block Diagram

12.1.2 Package

The Virtual Tape Library Profile doesn't stand alone. Figure 20 shows the component profiles that work with the Virtual Tape Library Profile to model a complete Virtual Library System product.

The objects in the center of Figure 20 represent the Virtual Library System Profile. The Virtual Library System uses specialized versions of the Generic Target Port and Generic initiator Port profile to model the ports.

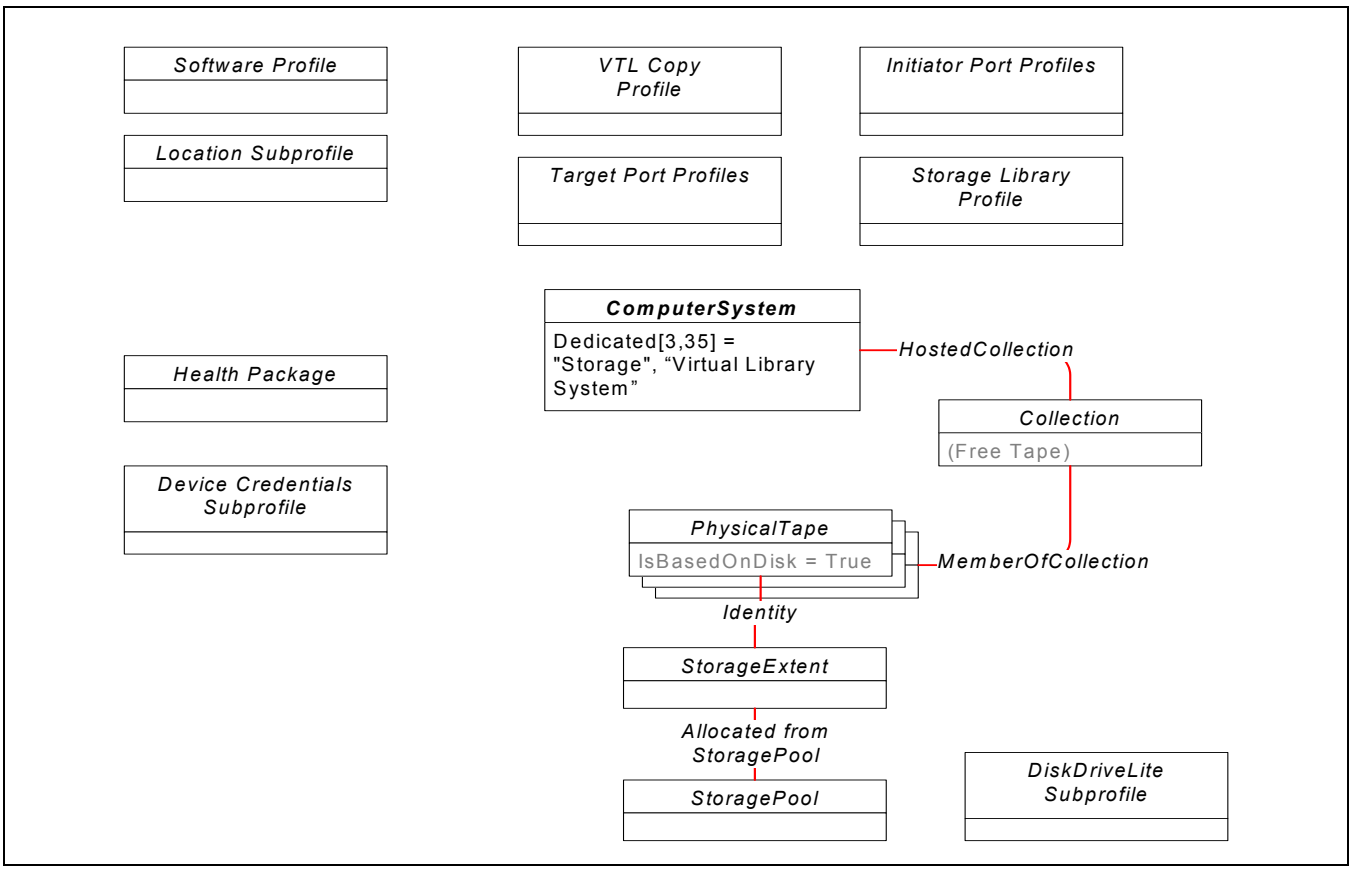

Figure 20 - Virtual Library System Package Diagram

12.1.3 Virtual Library System

12.1.3.1 Overview

Figure 21 shows the basic model of a Virtual Library System. This diagram does not contain all the classes and associations required to implement the profile but gives a picture of the main classes and associations as well as how they interact with major component profiles.

12.1.3.2 Virtual Library System ComputerSystem objects

The top-level system is modeled with **CIM_ComputerSystem;** the value of Dedicated includes 3 (Storage) and 35 (Virtual Library System). It shall be referenced by the CIM_ElementConformstoProfile association from the Profile Registration Profile. This object is also associated by CIM_SystemDevice to logical devices that are part of the Virtual Library System.

Virtual libraries shall have a CIM_ComputerSystem object with the Dedicated property including 3 (Storage) and 34 (Virtual Tape Library). The virtual library CIM_ComputerSystem object is associated by CIM_HostedDependancy. The box on the left of the Figure 21 contains the objects that represent a single Virtual Library. These classes shall be used for each Virtual Tape Library emulated by the system.

Physical libraries in the system shall have CIM_ComputerSystem objects with Dedicated property including values of 3 (Storage) and 22 (Media Library). The CIM_ComputerSystem objects are associated by CIM_ConcreteDependency. The physical library shall be modeled by the Storage Library Profile.

Disks may be modeled using the Disk Drive Lite profile. StorageExtent instances from Disk Drive Lite shall be associated to this profile's primordial StoragePool via ConcreteComponent. Storage from an array may also be used. In this case, each array LUN is modeled as a StorageExtent instance associated to this profile's primordial pool via ConcreteComponent.

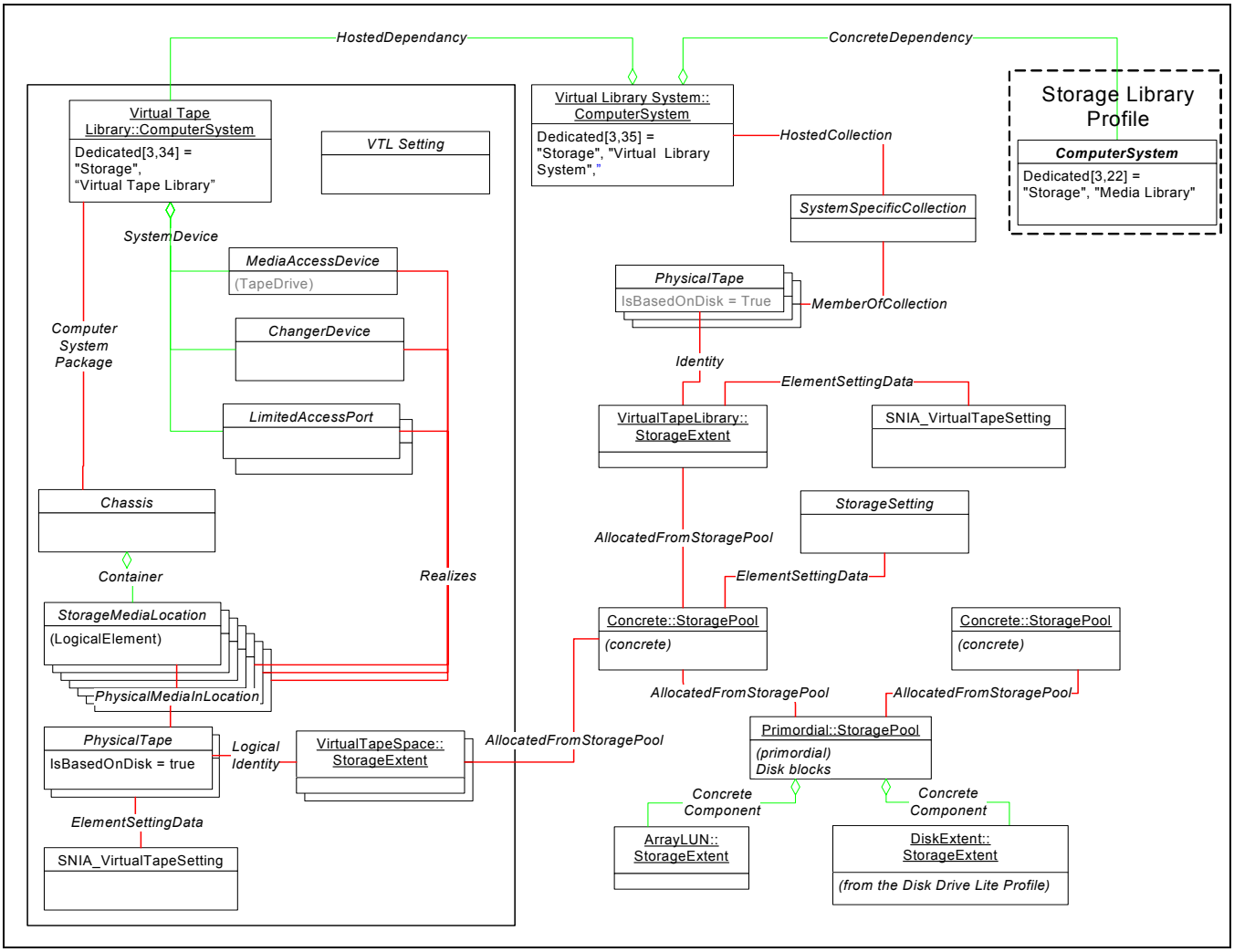

Figure 21 - Virtual Tape Library System

12.1.3.3 Block to Tape

Figure 22 details the objects involved in modeling the use of block storage to emulate virtual tapes.

The primordial CIM_StoragePool instances represent the block storage available in the Virtual Library System. Virtual tapes are in turn allocated from these pools. Virtual tapes are modeled by CIM_StorageExtents associated to the pools by CIM_AlocatedFromStoragePool. The virtual tape CIM_StorageExtents are also associated to CIM_PhysicalTape objects.

Imported logical units from disks or arrays is modeled as instances of StorageExtent associated to primordial storage pool. If this imported storage is from disks, the disks should be modeled using the Disk Drive Lite profile

with the StorageExtent associated to the primordial pool being the StorageExtent instance defined in the Disk Drive Lite profile.

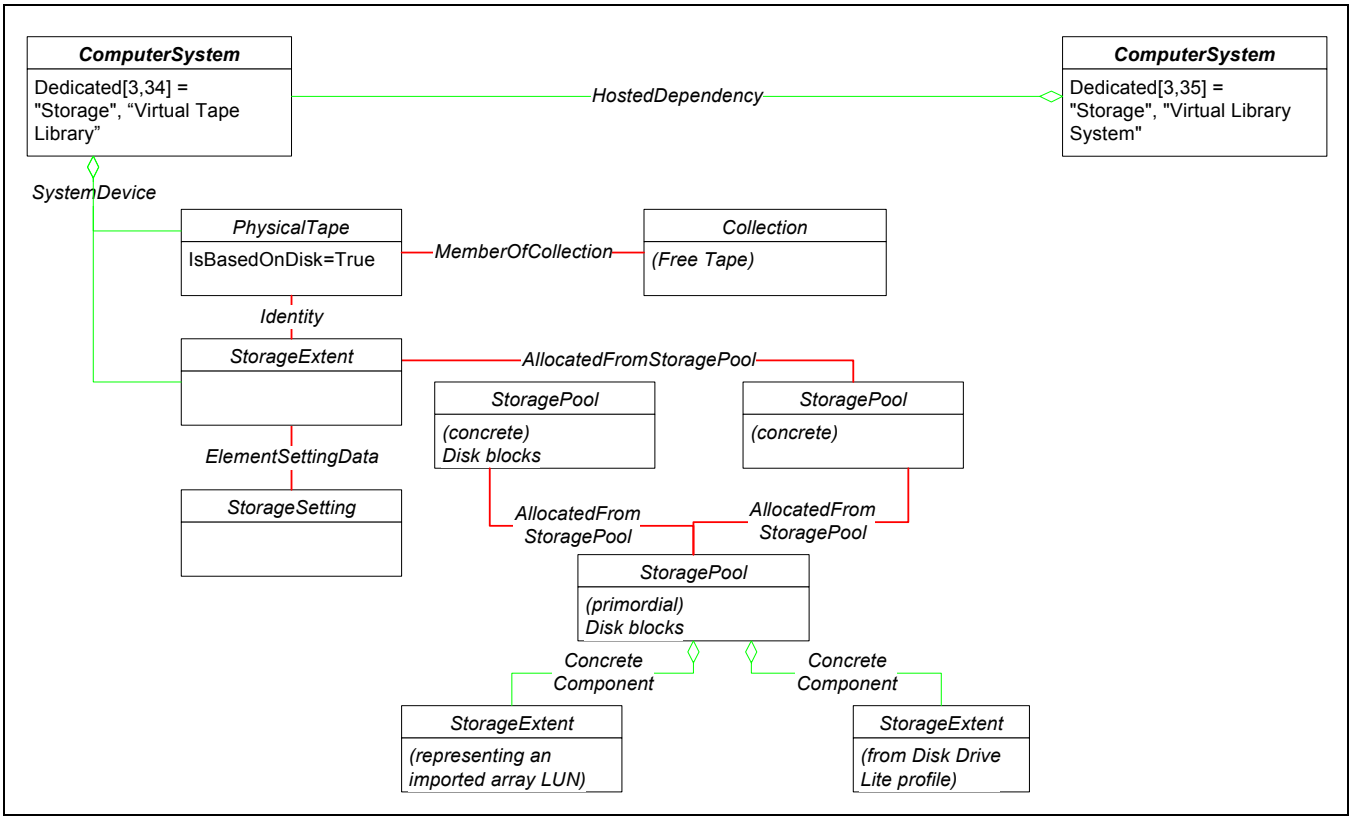

Figure 22 - VTL - Block to Tape

12.1.3.4 Virtual Library model

Virtual libraries shall have a CIM_ComputerSystem object with dedicated values of 3 (Storage) and 34 (Virtual Tape Library). The box on the left of the Figure 21 contains the objects that represent a single Virtual Library. These objects shall be replicated for each virtual Library emulated by the system. This CIM_ComputerSystem object shall scope the objects that are part of an instance. Logical devices that are part of the virtual library shall have CIM_SystemDevice associations back to the CMI_ComputerSystems object.

Each library shall have a CIM_Chassis, CIM_ChangerDevice, one or more CIM_MediaAccessDevice, one or more CIM_LimitedAccessPorts, and many CIM_StorageMediaLocation. These logical objects represent the virtual library the Virtual Library System is emulating.

The CIM_Chassis and the CIM_StorageMediaLocation objects represent the slots in a physical jukebox. They answer to inband and SMI-S move media commands as if the were physical slots.

There shall be one CIM_MediaAccessDevice object for each tape drive the virtual library is emulating. These objects shall be created and destroyed by the configuration commands described in Configuration of hardware (12.1.4.1) and assigned to ports by methods defined in Inband access (12.1.4.2)

CIM_LimitedAccessPorts may be able to eject the virtual media. Ejecting virtual media will cause a copy to physical media and then the ejection of the physical media.

12.1.3.5 Physical Library Model

Physical libraries in the system shall have CIM_ComputerSystem objects with dedicated values of 3 (Storage) and 22 (Media Library). The CIM_ComputerSystem objects are associated by CIM_ConcreteDependency. The physical library shall be modeled by the Storage Library Profile.

12.1.4 Virtual Library System configuration

The Virtual Library System model contains four main functions (Hardware Configuration, Virtual Library Configuration, Virtual Library management, Physical Library management).

12.1.4.1 Configuration of hardware

12.1.4.1.1 Services

The Virtual Library System Service class and Virtual Library System Capabilities class define methods used to configure the hardware of a Virtual Library System. The service contains the methods while the capabilities class contains properties that defines the methods and limits support by the implementation. Figure 23 shows the model for these classes.

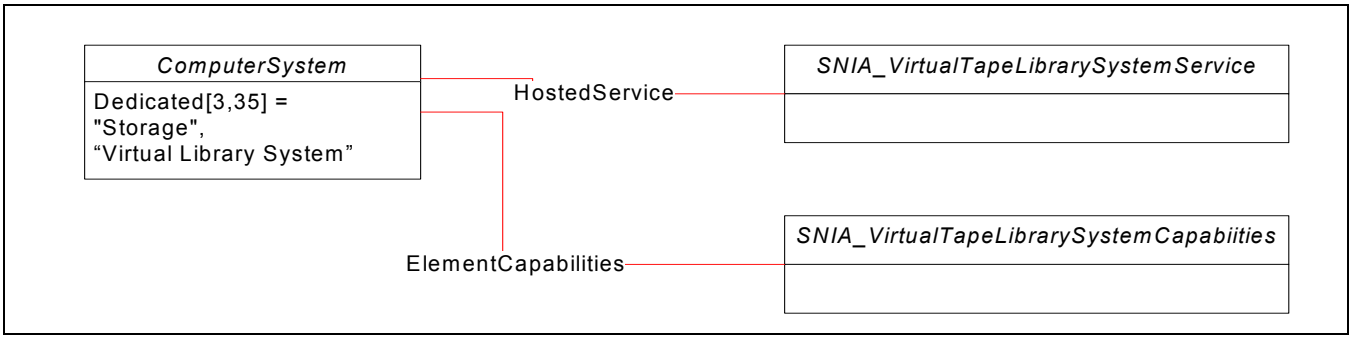

Figure 23 - Virtual Library System-Services

12.1.4.1.2 Array or Disk Configuration

A Virtual Library System uses block storage to hold images of virtual tapes. If the block storage comes from disks, the disks are modeled either using primordial StoragePool or Disk Drive Lite subprofile. If the block storage comes from RAID array systems, the imported LUNs are modeled as instances of StorageExtent associated to a primordial StoragePool.

RescanPhysicalHardware() causes the Virtual Library System to scan for external arrays and tape libraries.

12.1.4.1.3 Physical Tape Library configuration

Some Virtual Library System use physical tape libraries as storage for virtual tape libraries or as the destination of copy operations. The physical Library storage is modeled as either a direct attached storage media library. The SNIA_VirtualTapeLibrarySystemCapabilities class contains the following properties:

SupportsPhysicalLibrary is a uint32 that is set to a value of 2 (None) if the Virtual Library System does not have any physical library support or is set to a value of 3 (Local) if the Virtual Library System has a local library attached or a value of 4 (External) if a cascaded physical can be accessed. The rest of the properties are conditional on the SupportsPhysicalLibrary property being set to a value of either 3 or 4.

The SNIA_VirtualTapeLibrarySystemService class contains the following methods to attach storage media libraries:

ListPLibrary((out)LibraryList[]) is used to list potential tape libraries.

AttachPLibrary(Library) is conditional on the ExternalLibrary property being TRUE. The Library property is the “ID” of the library to attach. The ID is obtained from the ListLibrary() method.

DetachPLibrary(Library) removes access to an external library. The “Library” parameter is a REF to the CIM_ComputerSystem object for the library. NOTE: detaching a library stops all access to it and disconnects all associations to the Media Library model.

12.1.4.1.4 Port model

A Virtual Library System has multiple ports. These ports are used as targets (to provide service to a host) and/or as initiators (to communicate with external arrays and Physical Tape Libraries). The ports shall be modeled using the specializations of the Generic Initiator Port or Generic Target Port profiles. The CIM_logicalPort.UsageRestriction property shall be used to indicate the port usage.

The Virtual Library System service includes an optional method (SetPortUsage) to configure the usage of the ports. The Virtual Library SystemCababilities.ConfigPort property indicates if this method is supported.

12.1.4.2 Inband access

After the host facing ports (target ports) are defined, the inband access to virtual libraries, physical Libraries, and any other inband access is setup. Figure 24 is an instance diagram showing the model used to map/mask devices to the host facing ports.

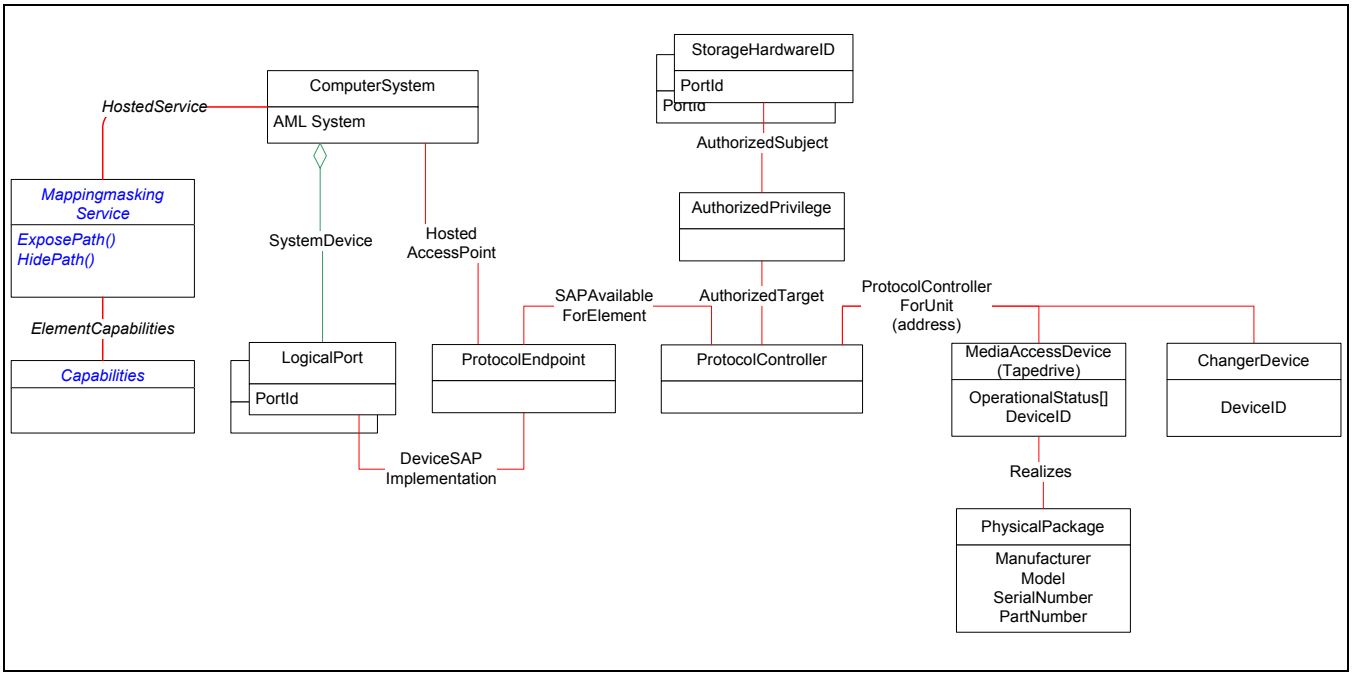

Figure 24 - Drive Mapping

12.1.4.3 Virtual Libraries Configuration

Figure 25 shows the part of the model related to the management of virtual libraries in a Virtual Tape Library System.

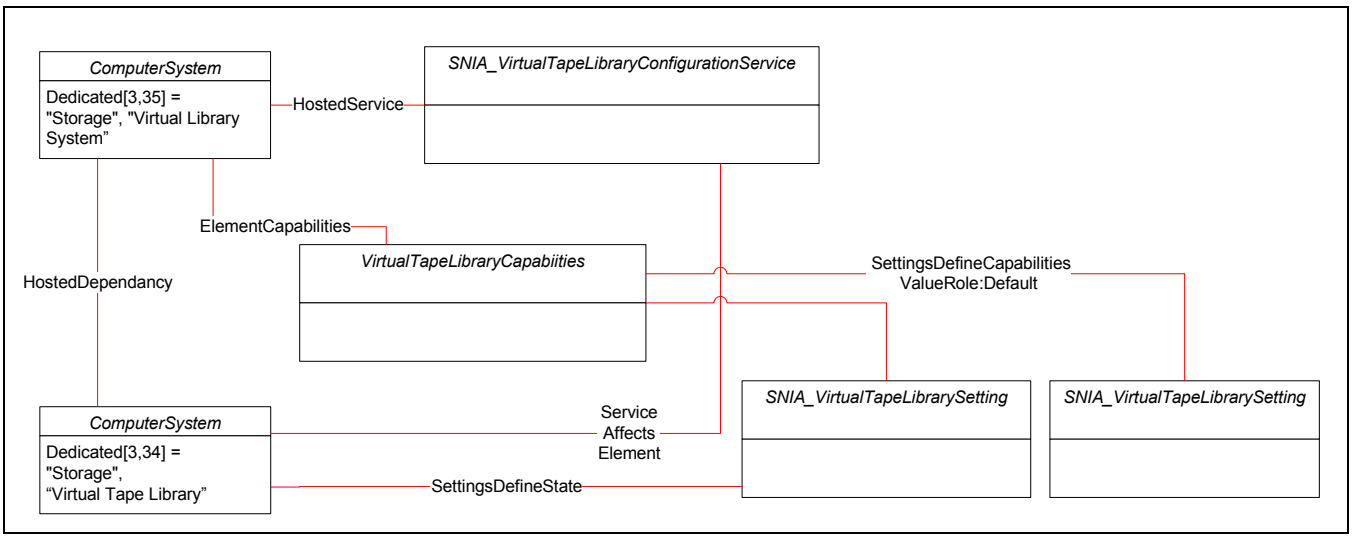

Figure 25 - Virtual Library Services

The SNIA_VirtualTapeLibraryConfigurationService class contains the following methods to manage Virtual media:

CreateLibrary(VirtualTapeLibrarySetting) is a required method. The method creates a virtual library using the information in the VTL setting object passed in. The base setting object is provided by the VTL service (canned). The object is copied and the variables are set. The object is then passed to this method. The VTL is created and the setting object is detached from the VTL service and attached to the VTL CIM_ComputerSystem object.

ModifyLibrary(VirtualTapeLibrarySetting)

ModifyLibrary is a required method. The method takes one parameter a REF to VirtualTapeLibrarySetting object associated to the VTL CIM_ComputerSystem object. The object contains a variable “Modify” that is an array containing a list of variables that may be modified.

Delete Library(Library, SaveTapes) Deletes a virtual library. The parameter “Library” is a REF to the CIM_ComputerSystem of the Virtual Tape Library. The virtual tapes in the slots will also be deleted and their storage returned to the pool.

12.1.4.4 Virtual Tape Service

Figure 26 shows the part of the model related to the management of virtual tapes in a virtual tape library.

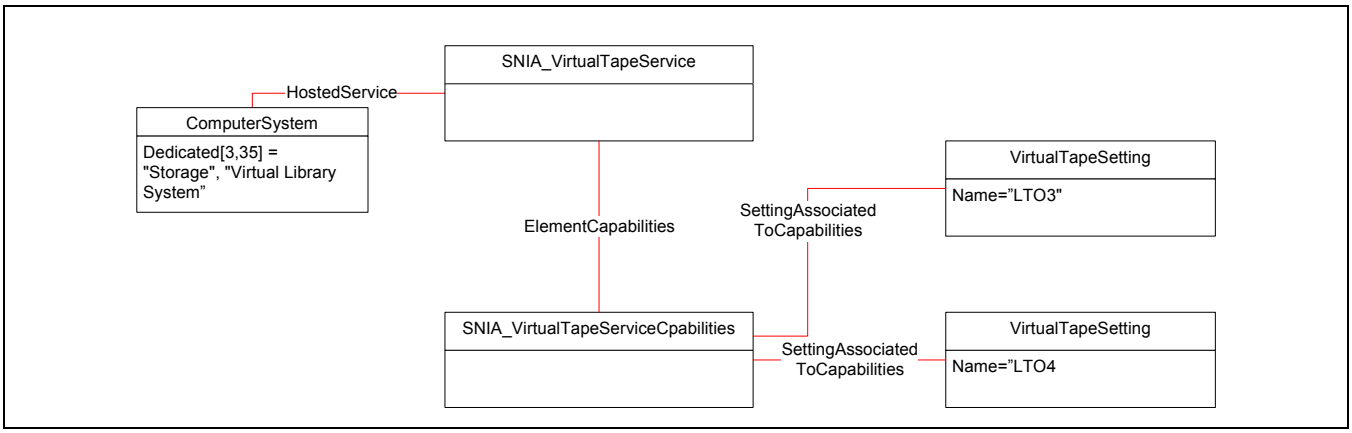

Figure 26 - Virtual Tape Service

The SNIA_VirtualTapeService class contains the following methods to manage Virtual media:

CreateTapeFromPool(Pool, Setting, Library, StartingLocation, Count) is required to create virtual media from available storage.

Pool is a reference to the Pool instance the media is to be allocated from.

Setting is a reference to a VirtualTapeLibrarySetting instance that defines the type of media being emulated.

Library is a reference to the CIM_ComputerSystem instance that represents the Virtual Tape Library.

StartingLocation is an integer with the slot number in it. New media will be put in this slot and higher numbered slots.

Count is an integer containing the number of media to be created.

ReturnTapeToPool(Tape) removes media from it's location and returns the storage to the Pool.

Tape is a reference to the CIM_PhysicalTape instance that represents the tape to be deleted.

MoveMedia(Source, Destination) moves virtual media from one slot to another.

Source is a reference to the CIM_StorageMediaLocation instance that represents the slot containing the virtual media.

Destination is a reference to the CIM_StorageMediaLocation instance that is the destination of the virtual media.

12.2 Health and Fault Management Consideration

Not supported in this version of the standard.

12.3 Cascading Considerations

Not supported in this version of the standard.

12.4 Supported Profiles and Packages

Table 77 describes the supported profiles for Virtual Tape Library.

Table 77 - Supported Profiles for Virtual Tape Library

Profile Name	Organization	Version	Requirement	Description
Disk Drive Lite	SNIA	1.4.0	Optional	
FC Target Ports	SNIA	1.4.0	Optional	
SAS Target Ports	SNIA	1.4.0	Optional	
SPI Target Ports	SNIA	1.4.0	Optional	
FC Initiator Ports	SNIA	1.4.0	Optional	
Health	SNIA	1.2.0	Mandatory	
Software	SNIA	1.4.0	Optional	
Storage Library	SNIA	1.2.0	Optional	
Indication	SNIA	1.4.0	Mandatory	
Multiple Computer System	SNIA	1.2.0	Optional	
Masking and Mapping	SNIA	1.4.0	Optional	
Virtual Tape Library Copy	SNIA	1.3.0	Optional	
Storage Server Asymmetry	SNIA	1.4.0	Optional	
Location	SNIA	1.4.0	Optional	

12.5 Methods of the profile

12.6 Client Considerations and Recipes

None.

12.7 Registered Name and Version

Virtual Tape Library version 1.3.0

12.8 CIM Elements

Table 78 describes the CIM elements for Virtual Tape Library.

Table 78 - CIM Elements for Virtual Tape Library

Element Name	Requirement	Description
12.8.1 CIM_AllocatedFromStoragePool (Pool from Concrete Pool)	Mandatory	AllocatedFromStoragePool.
12.8.2 CIM_AllocatedFromStoragePool (Pool from Primordial Pool)	Mandatory	AllocatedFromStoragePool.
12.8.3 CIM_AllocatedFromStoragePool (StorageExtent from Concrete Pool)	Mandatory	AllocatedFromStoragePool.
12.8.4 CIM_ChangerDevice	Optional	The media changer for a Virtual Tape Library.
12.8.5 CIM_Chassis (Virtual Library System)	Optional	The box for a Virtual Tape Library.
12.8.6 CIM_ComputerSystem (Virtual Library System)	Mandatory	'Top level' system that represents the entire Virtual Library System.
12.8.7 CIM_ComputerSystem (Virtual Tape Library)	Optional	'Top level' system that represents a Virtual Tape Library.
12.8.8 CIM_ComputerSystemPackage	Mandatory	This association links Chassis to the scoping system.
12.8.9 CIM_ConcreteComponent (StorageExtent from Primordial Pool)	Mandatory	ConcreteComponent.
12.8.10 CIM_ConcreteDependency (Virtual Library System to MediaLibrary)	Conditional	Conditional requirement: Support for SML profile. This association links the Virtual Library System ComputerSystem object to A MediaLibrary ComputerSystem objects.
12.8.11 CIM_Container (Chassis to slots)	Mandatory	This association links Slots to the chassis.
12.8.12 CIM_ElementCapabilities (Virtual Tape Library Capabilities)	Optional	
12.8.13 CIM_ElementCapabilities (Virtual Tape Library System Capabilities)	Optional	
12.8.14 CIM_ElementCapabilities (Virtual Tape Service Capabilities)	Optional	
12.8.15 CIM_ElementSettingData (Physical Tape)	Optional	
12.8.16 CIM_ElementSettingData (Pool Setting)	Optional	Associates StoragePool to StorageSetting.
12.8.17 CIM_HostedCollection	Optional	
12.8.18 CIM_HostedDependency (Virtual Library System to VirtualLibrary)	Mandatory	This association links the Virtual Library System ComputerSystem object to the VirtualLibrary ComputerSystem objects.

Table 78 - CIM Elements for Virtual Tape Library

Element Name	Requirement	Description
12.8.19 CIM_HostedService (Virtual Tape Library Configuration Service)	Optional	Associates the SNIA_VirtualTapeLibraryConfigurationService to the ComputerSystem representing the Virtual Library System.
12.8.20 CIM_HostedService (Virtual Tape Library System Service)	Optional	Associates the VirtualTapeLibrarySystemService to the ComputerSystem representing the Virtual Library System.
12.8.21 CIM_HostedService (Virtual Tape Service)	Optional	Associates the SNIA_VirtualTapeService to the ComputerSystem representing the Virtual Tape Library.
12.8.22 CIM_HostedStoragePool (Concrete)	Mandatory	
12.8.23 CIM_HostedStoragePool (Primordial)	Mandatory	
12.8.24 CIM_LimitedAccessPort	Optional	The media export port for a Virtual Tape Library.
12.8.25 CIM_LogicalIdentity	Mandatory	
12.8.26 CIM_MediaAccessDevice	Optional	The tapedrive for a Virtual Tape Library.
12.8.27 CIM_MemberOfCollection	Optional	
12.8.28 CIM_PhysicalMediaInLocation	Optional	This association links media to the slots.
12.8.29 CIM_Product	Optional	Asset information for the system.
12.8.30 CIM_ProductElementComponent (Virtual Library System)	Optional	
12.8.31 CIM_ProductElementComponent (Virtual Tape Library)	Optional	
12.8.32 CIM_Realizes (Slots to Changers)	Mandatory	This association links changers to the slots.
12.8.33 CIM_Realizes (Slots to Ports)	Mandatory	This association links ports to the slots.
12.8.34 CIM_Realizes (Slots to TapeDrive)	Mandatory	This association links drives to the slots.
12.8.35 CIM_ServiceAffectsElement	Mandatory	
12.8.36 CIM_SettingAssociatedToCapabilities	Mandatory	
12.8.37 CIM_SettingsDefineCapabilities	Mandatory	
12.8.38 CIM_SettingsDefineState	Mandatory	
12.8.39 CIM_StorageExtent (ArrayLUN)	Optional	The space used from the backing store.
12.8.40 CIM_StorageExtent (Virtual Tape Library)	Optional	Associated to virtual tape.
12.8.41 CIM_StorageMediaLocation	Optional	The slots and drive slots in a virtual tape library.

Table 78 - CIM Elements for Virtual Tape Library

Element Name	Requirement	Description
12.8.42 CIM_StoragePool (Concrete)	Mandatory	The concrete StoragePool. A concrete StoragePool shall be allocated from the Primordial StoragePool. It shall be used for allocating Virtual Tapes.
12.8.43 CIM_StoragePool (Primordial)	Mandatory	The pool of all storage available from the backing store.
12.8.44 CIM_StorageSetting	Mandatory	Properties of space allocated from the pool.
12.8.45 CIM_SystemDevice (System to Concrete StorageExtent)	Mandatory	This association links StorageExtents to the Virtual Tape Library ComputerSystem.
12.8.46 CIM_SystemDevice (System to Primordial StorageExtent)	Optional	This association links StorageExtents to the Virtual Library System.
12.8.47 CIM_SystemDevice (VTL to ChangerDevice)	Mandatory	This association links ChangerDevice to the scoping system.
12.8.48 CIM_SystemDevice (VTL to LimitedAccessPort)	Mandatory	This association links LimitedAccessDevice to the scoping system.
12.8.49 CIM_SystemDevice (VTL to MediaAccessDevice)	Mandatory	This association links MediaAccessDevice to the scoping system.
12.8.50 CIM_SystemSpecificCollection	Optional	Collection of unassigned virtual Tapes.
12.8.51 SNIA_PhysicalTape	Mandatory	The media in the Virtual Tape Library Collection.
12.8.52 SNIA_VirtualTapeLibraryCapabilities	Optional	Services used to create Virtual Media.
12.8.53 SNIA_VirtualTapeLibraryConfigurationService	Optional	Services used to create Virtual Tape Libraries.
12.8.54 SNIA_VirtualTapeLibrarySetting	Optional	Settings used to create the Virtual Tape Library.
12.8.55 SNIA_VirtualTapeLibrarySystemCapabilities	Optional	Features supported in VirtualTapeLibrarySystemService.
12.8.56 SNIA_VirtualTapeLibrarySystemService	Optional	Services used to set up the VTLS hardware.
12.8.57 SNIA_VirtualTapeService	Optional	Services used to create virtual media.
12.8.58 SNIA_VirtualTapeServiceCapabilities	Mandatory	
12.8.59 SNIA_VirtualTapeSetting	Optional	Setting for virtual Media.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_ComputerSystem AND ANY SourceInstance.CIM_ComputerSystem::Dedicated[*] = 34	Mandatory	CQL -Virtual Tape Library was created.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_PhysicalTape	Mandatory	Virtual Tape was created.

Table 78 - CIM Elements for Virtual Tape Library

Element Name	Requirement	Description
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_MediaAccessDevice	Mandatory	Virtual Tape Drive was created.
SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_ProtocolControllerForUnit	Mandatory	
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_ComputerSystem AND ANY SourceInstance.CIM_ComputerSystem::Dedicated[*] = 34	Mandatory	CQL -Virtual Tape Library was deleted.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_PhysicalTape	Mandatory	Virtual Tape was deleted.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_MediaAccessDevice	Mandatory	Virtual Tape drive was deleted from a Virtual Tape Library.
SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_ProtocolControllerForUnit	Mandatory	
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND ANY SourceInstance.CIM_ComputerSystem::Dedicated[*] = 34 AND SourceIsntance.CIM_ComputerSystem::OperationalStatus <> PreviousInstance.CIM_ComputerSystem::OperationalStatus	Mandatory	CQL -Status of a Virtual Tape Library has changed.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND ANY SourceInstance.CIM_ComputerSystem::Dedicated[*] = 35 AND SourceInstance.CIM_ComputerSystem::OperationalStatus <> PreviousInstance.CIM_ComputerSystem::OperationalStatus	Mandatory	CQL -Status of a Virtual Library System has changed.
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_ComputerSystem AND ANY SourceInstance.CIM_ComputerSystem::Dedicated[*] = 22 AND SourceInstance.CIM_ComputerSystem::OperationalStatus <> PreviousInstance.CIM_ComputerSystem::OperationalStatus	Mandatory	CQL -Status of an attached Media Library system has changed.

Table 78 - CIM Elements for Virtual Tape Library

Element Name	Requirement	Description
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_PhysicalTape	Mandatory	
SELECT * FROM CIM_InstModification WHERE SourceInstance ISA SNIA_VirtualTapeLibrarySetting	Mandatory	Indication that will identify when the settings of a Virtual Tape Library have changed.

12.8.1 CIM_AllocatedFromStoragePool (Pool from Concrete Pool)

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

Table 79 describes class CIM_AllocatedFromStoragePool (Pool from Concrete Pool).

Table 79 - SMI Referenced Properties/Methods for CIM_AllocatedFromStoragePool (Pool from Concrete Pool)

Properties	Flags	Requirement	Description & Notes
SpaceConsumed		Mandatory	
Antecedent		Mandatory	Antecedent references the parent pool from which the dependent pool is allocated.
Dependent		Mandatory	

12.8.2 CIM_AllocatedFromStoragePool (Pool from Primordial Pool)

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

Table 80 describes class CIM_AllocatedFromStoragePool (Pool from Primordial Pool).

Table 80 - SMI Referenced Properties/Methods for CIM_AllocatedFromStoragePool (Pool from Primordial Pool)

Properties	Flags	Requirement	Description & Notes
SpaceConsumed		Mandatory	

Table 80 - SMI Referenced Properties/Methods for CIM_AllocatedFromStoragePool (Pool from Primordial Pool)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	Antecedent references the parent pool from which the dependent pool is allocated.
Dependent		Mandatory	

12.8.3 CIM_AllocatedFromStoragePool (StorageExtent from Concrete Pool)

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

Table 81 describes class CIM_AllocatedFromStoragePool (StorageExtent from Concrete Pool).

Table 81 - SMI Referenced Properties/Methods for CIM_AllocatedFromStoragePool (StorageExtent from Concrete Pool)

Properties	Flags	Requirement	Description & Notes
SpaceConsumed		Mandatory	
Antecedent		Mandatory	Antecedent references the parent pool from which the dependent pool is allocated.
Dependent		Mandatory	

12.8.4 CIM_ChangerDevice

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic
Requirement: Optional

Table 82 describes class CIM_ChangerDevice.

Table 82 - SMI Referenced Properties/Methods for CIM_ChangerDevice

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

12.8.5 CIM_Chassis (Virtual Library System)

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic
Requirement: Optional

Table 83 describes class CIM_Chassis (Virtual Library System).

Table 83 - SMI Referenced Properties/Methods for CIM_Chassis (Virtual Library System)

Properties	Flags	Requirement	Description & Notes
Tag		Mandatory	
CreationClassName		Mandatory	
PackageType		Mandatory	Shall be 3 (ChassisFrame).
ChassisPackageType		Mandatory	
Manufacturer		Optional	
Model		Optional	
SerialNumber		Optional	
PartNumber		Optional	
SKU		Optional	
VendorCompatibilityStrings		Optional	
ElementName		Optional	

12.8.6 CIM_ComputerSystem (Virtual Library System)

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

Table 84 describes class CIM_ComputerSystem (Virtual Library System).

Table 84 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Virtual Library System)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Name		Mandatory	Unique identifier for the Virtual Library System. This should take the form of a string consisting of Vendor+Product+SerialNumber, derived from SCSI Inquiry Pages.
Dedicated		Mandatory	Indicates that this computer system is dedicated to operation as a Virtual Library System.
NameFormat		Mandatory	Format for Name property. Shall be 'HID'.
OperationalStatus		Mandatory	Overall status of the system.
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.
ElementName		Mandatory	User friendly name.
PrimaryOwnerContact	M	Optional	Contact details for Virtual Library System owner.
PrimaryOwnerName	M	Optional	Owner of the Virtual Library System.
OtherIdentifyingInfo		Optional	Other data that could be used to identify the Virtual Library System.
IdentifyingDescriptions		Optional	Provides explanations and details for the entries in the OtherIdentifyingInfo property.

12.8.7 CIM_ComputerSystem (Virtual Tape Library)

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic
Requirement: Optional

Table 85 describes class CIM_ComputerSystem (Virtual Tape Library).

Table 85 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Virtual Tape Library)

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Name		Mandatory	
Dedicated		Mandatory	Indicates that this computer system is dedicated to operation as a Virtual Tape Library.
NameFormat		Mandatory	Format for Name property.

Table 85 - SMI Referenced Properties/Methods for CIM_ComputerSystem (Virtual Tape Library)

Properties	Flags	Requirement	Description & Notes
OperationalStatus		Mandatory	Overall status of the system.
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.
ElementName		Mandatory	User friendly name.
PrimaryOwnerContact	M	Optional	Contact details for Virtual Tape Library owner.
PrimaryOwnerName	M	Optional	Owner of the Virtual Tape Library.
OtherIdentifyingInfo		Optional	Other data that could be used to identify the Virtual Tape Library.
IdentifyingDescriptions		Optional	Provides explanations and details for the entries in the OtherIdentifyingInfo property.

12.8.8 CIM_ComputerSystemPackage

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

Table 86 describes class CIM_ComputerSystemPackage.

Table 86 - SMI Referenced Properties/Methods for CIM_ComputerSystemPackage

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

12.8.9 CIM_ConcreteComponent (StorageExtent from Primordial Pool)

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

Table 87 describes class CIM_ConcreteComponent (StorageExtent from Primordial Pool).

Table 87 - SMI Referenced Properties/Methods for CIM_ConcreteComponent (StorageExtent from Primordial Pool)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	Antecedent references the parent pool from which the dependent pool is allocated.
PartComponent		Mandatory	

12.8.10 CIM_ConcreteDependency (Virtual Library System to MediaLibrary)

Created By: Static
Modified By: Static
Deleted By: Static
Requirement: Support for SML profile.

Table 88 describes class CIM_ConcreteDependency (Virtual Library System to MediaLibrary).

Table 88 - SMI Referenced Properties/Methods for CIM_ConcreteDependency (Virtual Library System to MediaLibrary)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	Virtual Library System ComputerSystem object.
Dependent		Mandatory	Storage Library ComputerSystem object.

12.8.11 CIM_Container (Chassis to slots)

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

Table 89 describes class CIM_Container (Chassis to slots).

Table 89 - SMI Referenced Properties/Methods for CIM_Container (Chassis to slots)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

12.8.12 CIM_ElementCapabilities (Virtual Tape Library Capabilities)

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

Table 90 describes class CIM_ElementCapabilities (Virtual Tape Library Capabilities).

Table 90 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (Virtual Tape Library Capabilities)

Properties	Flags	Requirement	Description & Notes
Capabilities		Mandatory	Reference to VirtualTapeLibraryCapabilities.
ManagedElement		Mandatory	Reference to Virtual Library System ComputerSystem.

12.8.13 CIM_ElementCapabilities (Virtual Tape Library System Capabilities)

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

Table 91 describes class CIM_ElementCapabilities (Virtual Tape Library System Capabilities).

Table 91 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (Virtual Tape Library System Capabilities)

Properties	Flags	Requirement	Description & Notes
Capabilities		Mandatory	Reference to SNIA_VirtualTapeLibrarySystemCapabilities.
ManagedElement		Mandatory	Reference to the Virtual Library System.

12.8.14 CIM_ElementCapabilities (Virtual Tape Service Capabilities)

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

Table 92 describes class CIM_ElementCapabilities (Virtual Tape Service Capabilities).

Table 92 - SMI Referenced Properties/Methods for CIM_ElementCapabilities (Virtual Tape Service Capabilities)

Properties	Flags	Requirement	Description & Notes
Capabilities		Mandatory	Reference to VirtualTapeServiceCapabilities.
ManagedElement		Mandatory	Reference to VirtualTapeService.

12.8.15 CIM_ElementSettingData (Physical Tape)

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

Table 93 describes class CIM_ElementSettingData (Physical Tape).

Table 93 - SMI Referenced Properties/Methods for CIM_ElementSettingData (Physical Tape)

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	
SettingData		Mandatory	

12.8.16 CIM_ElementSettingData (Pool Setting)

Associates StoragePool to StorageSetting.

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

Table 94 describes class CIM_ElementSettingData (Pool Setting).

Table 94 - SMI Referenced Properties/Methods for CIM_ElementSettingData (Pool Setting)

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	Reference to StoragePool.
SettingData		Mandatory	Reference to StorageSetting.

12.8.17 CIM_HostedCollection

Created By: Static
Modified By: Static

Deleted By: Static
Requirement: Optional

Table 95 describes class CIM_HostedCollection.

Table 95 - SMI Referenced Properties/Methods for CIM_HostedCollection

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

12.8.18 CIM_HostedDependency (Virtual Library System to VirtualLibrary)

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

Table 96 describes class CIM_HostedDependency (Virtual Library System to VirtualLibrary).

Table 96 - SMI Referenced Properties/Methods for CIM_HostedDependency (Virtual Library System to VirtualLibrary)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	Virtual Library System ComputerSystem object.
Dependent		Mandatory	VirtualLibrary ComputerSystem object.

12.8.19 CIM_HostedService (Virtual Tape Library Configuration Service)

Associates the SNIA_VirtualTapeLibraryConfigurationService to the ComputerSystem representing the Virtual Library System.

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

Table 97 describes class CIM_HostedService (Virtual Tape Library Configuration Service).

Table 97 - SMI Referenced Properties/Methods for CIM_HostedService (Virtual Tape Library Configuration Service)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	The reference to the ComputerSystem representing the Virtual Library System.
Dependent		Mandatory	The reference to the SNIA_VirtualTapeLibraryConfigurationService.

12.8.20 CIM_HostedService (Virtual Tape Library System Service)

Associates the VirtualTapeLibrarySystemService to the ComputerSystem representing the Virtual Library System.

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

Table 98 describes class CIM_HostedService (Virtual Tape Library System Service).

Table 98 - SMI Referenced Properties/Methods for CIM_HostedService (Virtual Tape Library System Service)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	The reference to the ComputerSystem representing the Virtual Library System.
Dependent		Mandatory	The reference to the SNIA_VirtualTapeLibrarySystemService.

12.8.21 CIM_HostedService (Virtual Tape Service)

Associates the SNIA_VirtualTapeService to the ComputerSystem representing the Virtual Tape Library.

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

Table 99 describes class CIM_HostedService (Virtual Tape Service).

Table 99 - SMI Referenced Properties/Methods for CIM_HostedService (Virtual Tape Service)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	The reference to the ComputerSystem representing the Virtual Tape Library.
Dependent		Mandatory	The reference to the SNIA_VirtualTapeService.

12.8.22 CIM_HostedStoragePool (Concrete)

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

Table 100 describes class CIM_HostedStoragePool (Concrete).

Table 100 - SMI Referenced Properties/Methods for CIM_HostedStoragePool (Concrete)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

12.8.23 CIM_HostedStoragePool (Primordial)

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

Table 101 describes class CIM_HostedStoragePool (Primordial).

Table 101 - SMI Referenced Properties/Methods for CIM_HostedStoragePool (Primordial)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

12.8.24 CIM_LimitedAccessPort

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic
Requirement: Optional

Table 102 describes class CIM_LimitedAccessPort.

Table 102 - SMI Referenced Properties/Methods for CIM_LimitedAccessPort

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
DeviceID		Mandatory	
OperationalStatus		Optional	
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.

12.8.25 CIM_LogicalIdentity

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

Table 103 describes class CIM_LogicalIdentity.

Table 103 - SMI Referenced Properties/Methods for CIM_LogicalIdentity

Properties	Flags	Requirement	Description & Notes
SystemElement		Mandatory	
SameElement		Mandatory	

12.8.26 CIM_MediaAccessDevice

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic
Requirement: Optional

Table 104 describes class CIM_MediaAccessDevice.

Table 104 - SMI Referenced Properties/Methods for CIM_MediaAccessDevice

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
DeviceID		Mandatory	
OperationalStatus		Optional	
StatusDescriptions		Optional	Additional information related to the values in OperationalStatus.
NeedsCleaning		Optional	Shall be false for virtual drives.
MountCount		Optional	

12.8.27 CIM_MemberOfCollection

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

Table 105 describes class CIM_MemberOfCollection.

Table 105 - SMI Referenced Properties/Methods for CIM_MemberOfCollection

Properties	Flags	Requirement	Description & Notes
Collection		Mandatory	
Member		Mandatory	

12.8.28 CIM_PhysicalMediaInLocation

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

Table 106 describes class CIM_PhysicalMediaInLocation.

Table 106 - SMI Referenced Properties/Methods for CIM_PhysicalMediaInLocation

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

12.8.29 CIM_Product

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

Table 107 describes class CIM_Product.

Table 107 - SMI Referenced Properties/Methods for CIM_Product

Properties	Flags	Requirement	Description & Notes
ElementName		Mandatory	
Name		Mandatory	
IdentifyingNumber		Mandatory	
Vendor		Mandatory	
Version		Mandatory	

12.8.30 CIM_ProductElementComponent (Virtual Library System)

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

Table 108 describes class CIM_ProductElementComponent (Virtual Library System).

Table 108 - SMI Referenced Properties/Methods for CIM_ProductElementComponent (Virtual Library System)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

12.8.31 CIM_ProductElementComponent (Virtual Tape Library)

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

Table 109 describes class CIM_ProductElementComponent (Virtual Tape Library).

Table 109 - SMI Referenced Properties/Methods for CIM_ProductElementComponent (Virtual Tape Library)

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	
PartComponent		Mandatory	

12.8.32 CIM_Realizes (Slots to Changers)

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

Table 110 describes class CIM_Realizes (Slots to Changers).

Table 110 - SMI Referenced Properties/Methods for CIM_Realizes (Slots to Changers)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

12.8.33 CIM_Realizes (Slots to Ports)

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

Table 111 describes class CIM_Realizes (Slots to Ports).

Table 111 - SMI Referenced Properties/Methods for CIM_Realizes (Slots to Ports)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

12.8.34 CIM_Realizes (Slots to TapeDrive)

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

Table 112 describes class CIM_Realizes (Slots to TapeDrive).

Table 112 - SMI Referenced Properties/Methods for CIM_Realizes (Slots to TapeDrive)

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

12.8.35 CIM_ServiceAffectsElement

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

Table 113 describes class CIM_ServiceAffectsElement.

Table 113 - SMI Referenced Properties/Methods for CIM_ServiceAffectsElement

Properties	Flags	Requirement	Description & Notes
AffectedElement		Mandatory	Reference to the Virtual Tape Library ComputerSystem.
AffectingElement		Mandatory	

12.8.36 CIM_SettingAssociatedToCapabilities

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

Requirement: Mandatory

Table 114 describes class CIM_SettingAssociatedToCapabilities.

Table 114 - SMI Referenced Properties/Methods for CIM_SettingAssociatedToCapabilities

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	Reference to VirtualTapeServiceCapabilities.
Dependent		Mandatory	Reference to VirtualTapeSetting.

12.8.37 CIM_SettingsDefineCapabilities

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

Table 115 describes class CIM_SettingsDefineCapabilities.

Table 115 - SMI Referenced Properties/Methods for CIM_SettingsDefineCapabilities

Properties	Flags	Requirement	Description & Notes
GroupComponent		Mandatory	Reference to VirtualTapeLibraryCapabilities.
PartComponent		Mandatory	Reference to VirtualTapeLibrarySetting.

12.8.38 CIM_SettingsDefineState

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

Table 116 describes class CIM_SettingsDefineState.

Table 116 - SMI Referenced Properties/Methods for CIM_SettingsDefineState

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	
SettingData		Mandatory	

12.8.39 CIM_StorageExtent (ArrayLUN)

Created By: Extrinsic

Modified By: Extrinsic
Deleted By: Extrinsic
Requirement: Optional

Table 117 describes class CIM_StorageExtent (ArrayLUN).

Table 117 - SMI Referenced Properties/Methods for CIM_StorageExtent (ArrayLUN)

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
DeviceID		Mandatory	
DataOrganization		Mandatory	Fixed Block (disk) or Variable Block (tape).
Primordial		Mandatory	Shall be 'true'.
NumberOfBlocks		Mandatory	Used with block size.
BlockSize		Mandatory	
ExtentStatus		Mandatory	
OperationalStatus		Mandatory	

12.8.40 CIM_StorageExtent (Virtual Tape Library)

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic
Requirement: Optional

Table 118 describes class CIM_StorageExtent (Virtual Tape Library).

Table 118 - SMI Referenced Properties/Methods for CIM_StorageExtent (Virtual Tape Library)

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
SystemName		Mandatory	
CreationClassName		Mandatory	
DeviceID		Mandatory	
DataOrganization		Mandatory	Fixed Block (disk) or Variable Block (tape).
Primordial		Mandatory	Shall be 'false'.

Table 118 - SMI Referenced Properties/Methods for CIM_StorageExtent (Virtual Tape Library)

Properties	Flags	Requirement	Description & Notes
NumberOfBlocks		Mandatory	Used with block size.
BlockSize		Mandatory	
ExtentStatus		Mandatory	
OperationalStatus		Mandatory	

12.8.41 CIM_StorageMediaLocation

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic
Requirement: Optional

Table 119 describes class CIM_StorageMediaLocation.

Table 119 - SMI Referenced Properties/Methods for CIM_StorageMediaLocation

Properties	Flags	Requirement	Description & Notes
CreationClassName		Mandatory	
Tag		Mandatory	
LocationType		Mandatory	Slot, MediaAccessDevice, or Limited Access Port.
LocationCoordinates		Mandatory	
MediaTypesSupported		Mandatory	
MediaCapacity		Mandatory	

12.8.42 CIM_StoragePool (Concrete)

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

Table 120 describes class CIM_StoragePool (Concrete).

Table 120 - SMI Referenced Properties/Methods for CIM_StoragePool (Concrete)

Properties	Flags	Requirement	Description & Notes
Primordial		Mandatory	Shall be false.
InstanceID		Mandatory	
ElementName		Optional	
PoolID		Mandatory	A unique name in the context of this system that identifies this Pool.
TotalManagedSpace		Mandatory	
RemainingManaged Space		Mandatory	

12.8.43 CIM_StoragePool (Primordial)

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

Table 121 describes class CIM_StoragePool (Primordial).

Table 121 - SMI Referenced Properties/Methods for CIM_StoragePool (Primordial)

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
PoolID		Mandatory	A unique name in the context of this system that identifies this Pool.
Primordial		Mandatory	Shall be 'true'.
TotalManagedSpace		Mandatory	
RemainingManaged Space		Mandatory	
ElementName		Optional	

12.8.44 CIM_StorageSetting

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

Table 122 describes class CIM_StorageSetting.

Table 122 - SMI Referenced Properties/Methods for CIM_StorageSetting

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
NoSinglePointOfFailure		Mandatory	
ElementName		Optional	

12.8.45 CIM_SystemDevice (System to Concrete StorageExtent)

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

Table 123 describes class CIM_SystemDevice (System to Concrete StorageExtent).

Table 123 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to Concrete StorageExtent)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	Reference to StorageExtent.
GroupComponent		Mandatory	Reference to Virtual Tape Library ComputerSystem.

12.8.46 CIM_SystemDevice (System to Primordial StorageExtent)

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

Table 124 describes class CIM_SystemDevice (System to Primordial StorageExtent).

Table 124 - SMI Referenced Properties/Methods for CIM_SystemDevice (System to Primordial StorageExtent)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	Reference to StorageExtent.
GroupComponent		Mandatory	Reference to Virtil Library System ComputerSystem.

12.8.47 CIM_SystemDevice (VTL to ChangerDevice)

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

Table 125 describes class CIM_SystemDevice (VTL to ChangerDevice).

Table 125 - SMI Referenced Properties/Methods for CIM_SystemDevice (VTL to ChangerDevice)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

12.8.48 CIM_SystemDevice (VTL to LimitedAccessPort)

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

Table 126 describes class CIM_SystemDevice (VTL to LimitedAccessPort).

Table 126 - SMI Referenced Properties/Methods for CIM_SystemDevice (VTL to LimitedAccess-Port)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

12.8.49 CIM_SystemDevice (VTL to MediaAccessDevice)

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

Table 127 describes class CIM_SystemDevice (VTL to MediaAccessDevice).

Table 127 - SMI Referenced Properties/Methods for CIM_SystemDevice (VTL to MediaAccessDevice)

Properties	Flags	Requirement	Description & Notes
PartComponent		Mandatory	
GroupComponent		Mandatory	

12.8.50 CIM_SystemSpecificCollection

Collection of unassigned virtual Tapes.

Created By: Extrinsic
Modified By: Extrinsic
Deleted By: Extrinsic
Requirement: Optional

Table 128 describes class CIM_SystemSpecificCollection.

Table 128 - SMI Referenced Properties/Methods for CIM_SystemSpecificCollection

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
ElementName		Optional	

12.8.51 SNIA_PhysicalTape

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

Table 129 describes class SNIA_PhysicalTape.

Table 129 - SMI Referenced Properties/Methods for SNIA_PhysicalTape

Properties	Flags	Requirement	Description & Notes
IsBasedOnDisk		Mandatory	Shall be 'true'.
Usage		Optional	Used by Virtual Tape Library Copy profile to indicate whether a tape is currently involved in a copy operation.
OtherUsageDescription		Optional	If the 'Usage' property is present and has a value of 'Other', this property provides additional detail and explanation for the current Usage state.

12.8.52 SNIA_VirtualTapeLibraryCapabilities

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

Table 130 describes class SNIA_VirtualTapeLibraryCapabilities.

Table 130 - SMI Referenced Properties/Methods for SNIA_VirtualTapeLibraryCapabilities

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
LibraryTypesSupported		Mandatory	
MaxVTLsSupported		Optional	
MaxDrivesSupported		Optional	
MaxAccessPortsSupported		Optional	
IsThinTapeSupported		Mandatory	

12.8.53 SNIA_VirtualTapeLibraryConfigurationService

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

Table 131 describes class SNIA_VirtualTapeLibraryConfigurationService.

Table 131 - SMI Referenced Properties/Methods for SNIA_VirtualTapeLibraryConfigurationService

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
Name		Mandatory	
CreateVTL()		Optional	Creates a new Virtual Library.

Table 131 - SMI Referenced Properties/Methods for SNIA_VirtualTapeLibraryConfigurationService

Properties	Flags	Requirement	Description & Notes
ModifyVTL()		Optional	Modifies the configurable settings of a Virtual Library.
DeleteVTL()		Optional	Deletes a Virtual Library.

12.8.54 SNIA_VirtualTapeLibrarySetting

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

Table 132 describes class SNIA_VirtualTapeLibrarySetting.

Table 132 - SMI Referenced Properties/Methods for SNIA_VirtualTapeLibrarySetting

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
LibraryType		Mandatory	
LibraryName		Optional	If present, this shall be the name of the Virtual Library associated with these settings.
SlotCount		Optional	If present, this shall be the number of slots in the Virtual Library associated with these settings.
IsThinTape		Mandatory	
ThinTapeSize		Mandatory	
MaxTapeSize		Mandatory	
DriveType		Mandatory	
DriveCount		Optional	If present, this shall be the number of drives in the Virtual Library associated with these settings.
DriveNames		Optional	
TapeBarcodeRange		Optional	
Modify		Optional	Property used when modifying the settings of an existing Virtual Tape Library.

12.8.55 SNIA_VirtualTapeLibrarySystemCapabilities

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

Requirement: Optional

Table 133 describes class SNIA_VirtualTapeLibrarySystemCapabilities.

Table 133 - SMI Referenced Properties/Methods for SNIA_VirtualTapeLibrarySystemCapabilities

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
ConfigPort		Mandatory	
ExternalPhysicalLibrary		Mandatory	

12.8.56 SNIA_VirtualTapeLibrarySystemService

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

Table 134 describes class SNIA_VirtualTapeLibrarySystemService.

Table 134 - SMI Referenced Properties/Methods for SNIA_VirtualTapeLibrarySystemService

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
Name		Mandatory	
SetPortUse()		Optional	Set the port to 'Initiator' or 'Target'.
ListPLibrary()		Optional	Finds the Physical Libraries connected to the Virtual Library System.
AttachPLibrary()		Optional	Allows a Physical Library to be used by the Virtual Library System for copy and export operations.
DetachPLibrary()		Optional	Removes a Physical Library from the context of a Virtual Library System, disallowing copy and export operations to that library.
RescanPhysicalHardware()		Optional	Scans for external Physical Libraries and Block Storage Arrays.

12.8.57 SNIA_VirtualTapeService

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

Table 135 describes class SNIA_VirtualTapeService.

Table 135 - SMI Referenced Properties/Methods for SNIA_VirtualTapeService

Properties	Flags	Requirement	Description & Notes
SystemCreationClassName		Mandatory	
CreationClassName		Mandatory	
SystemName		Mandatory	
Name		Mandatory	
CreateTapeFromPool()		Optional	Creates one or more virtual tapes in a Virtual Library.
DeleteTape()		Optional	Deletes a virtual tape from a Virtual Library.
MoveMedia()		Optional	Moves a virtual tape from one location to another.

12.8.58 SNIA_VirtualTapeServiceCapabilities

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

Table 136 describes class SNIA_VirtualTapeServiceCapabilities.

Table 136 - SMI Referenced Properties/Methods for SNIA_VirtualTapeServiceCapabilities

Properties	Flags	Requirement	Description & Notes
SupportedTypes		Mandatory	
SupportedMethods	N	Mandatory	

12.8.59 SNIA_VirtualTapeSetting

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

Table 137 describes class SNIA_VirtualTapeSetting.

Table 137 - SMI Referenced Properties/Methods for SNIA_VirtualTapeSetting

Properties	Flags	Requirement	Description & Notes
InstanceID		Mandatory	
Type		Mandatory	
ElementName		Optional	

EXPERIMENTAL

EXPERIMENTAL

Clause 13: Virtual Tape Library Copy Profile

13.1 Description

13.1.1 Overview

This profile describes a backup application independent way of triggering tape copies in SMI-S.

Research shows that backup applications need to add metadata to tape when triggering tape copies. Our goal is to propose an interface that will allow backup applications to write and read this metadata to and from tape at any time during the tape copy process.

13.2 Tape Copy Services

13.2.1 Summary

Figure 27 summarizes available tape copy services.

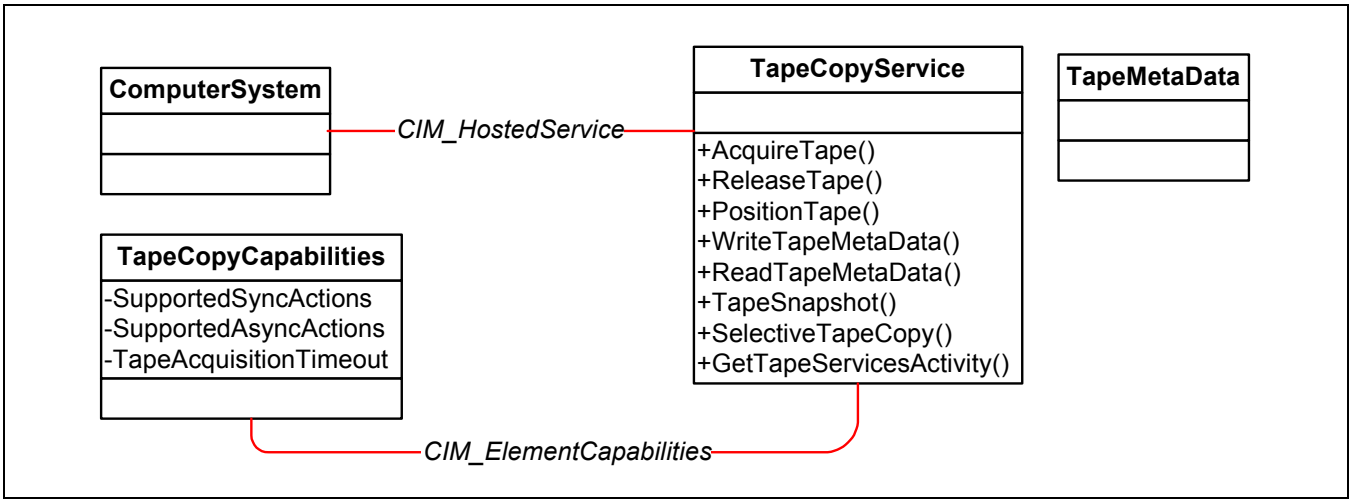

Figure 27 - Tape Copy Services Class Diagram

Tape Copy Services introduces the concept of acquiring and releasing tapes. Any sequence of position, read or write operations described further shall be performed within these two calls. These specifications do not specify provider behavior in the event of concurrent access from multiple clients (See Concurrency Considerations (13.2.2.3)). However, AcquireTape shall return an error if the tape is currently acquired. The activity status of a given tape should be obtained via a call to GetTapeServicesActivity() in order to check whether it is safe to call AcquireTape. Note that two calls to AcquireTape will be necessary for tape copy operations (e.g., TapeSnapshot or SelectiveTapeCopy…): one for the source tape and another one for the destination tape.

13.2.2 Definitions

13.2.2.1 TapeMetaData Class

The TapeMetaData class shown in Figure 28 is used to represent metadata and is composed of:

- "An array of strings encoded in hex binary, using the Octetstring qualifier

- "An integer value as to the number of file marks to write before writing metadata.
- "An integer value as to the number of file marks to write after writing metadata.
- "An array of integers or "bit codes" that indicate the block size of each data string.

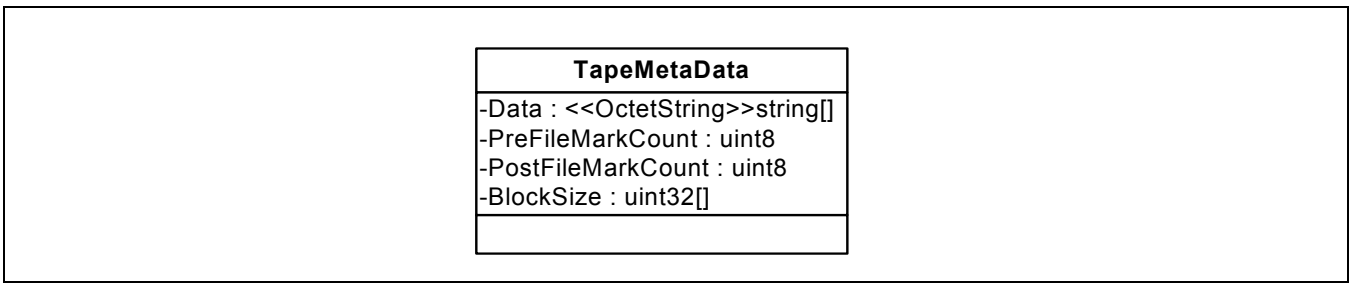

Figure 28 - TapeMetaData Class Definition

Data is defined as a string array and qualified by Octetstring. This means that every string will be encoded using the cim:cimHexBinary type. In this encoding scheme, every byte of data is encoded in 4 hexadecimal characters leading to a 4:1 negative compression ratio, accounting for the fact that CIM uses the UCS2 character set. This format allows the TapeMetaData object to be packaged as an embedded object (See definition of WriteTapeMetaData in WriteTapeMetaData (13.2.2.7))

13.2.2.2 Considerations on Load/Unload

This interface purposely does not define load and unload calls in order to allow for tape copy logic to reside in the device itself. For instance, the device has unique knowledge of what drives should be used to trigger a copy.

However, upon processing an AcquireTape call, the provider shall ensure that a tape is loaded and ready for data access, at least until ReleaseTape is called. AcquireTape may or may not result in a tape being loaded depending on whether the tape was already accessible.

The unload behavior is undefined. These specifications make no recommendation as to whether a tape should be unloaded after a ReleaseTape call is processed. Some implementations may decide to unload the tape immediately, after an arbitrarily defineTape Met a Dataimeout has expired, or simply when the drive is required for another task.

13.2.2.3 Concurrency Considerations

It is beyond the scope of this profile to specify concurrency behavior and/or to define a locking mechanism associated to the action of acquiring a tape. However, AcquireTape will fail if called twice (from any client) without an intermediate ReleaseTape. Likewise, ReleaseTape will return an error if called on a tape that was not previously acquired. The activity/status of a given tape can be obtained via the GetTapeServicesActivity operation.

13.2.2.4 Acquire Tape

AcquireTape initializes a 'transaction' for a given tape and informs the provider that a sequence of actions will be performed on that tape. A call to AcquireTape is required before any sequence of actions can be performed, actions that will result in an error if the tape activity isn't "Acquired" (See GetTapeCopyServicesActivity (13.2.2.11) for defined activity values). These actions are:

"PositionTape

"WriteMetaData

"ReadMetaData

"TapeSnapshot

"SelectiveTapeCopy

GetTapeServicesActivity is the only call pertaining to a tape that doesn't require that tape to be "Acquired"

For any given tape, additional calls to AcquireTape shall return an error unless ReleaseTape was called or if the AcquireTape timeout has expired.

AcquireTape may or may not load a tape (See Considerations on Load/Unload (13.2.2.2))

AcquireTape (dest,timeout, Job)

"dest [IN:CIM_PhysicalTape REF]: destination tape

"timeout[IN:datetime]: Timeout after which an inactive 'transaction' will be released automatically by the provider. This argument is optional: if not specified, the default timeout specified by TapeCopyCapabilities.TapeAcquisitionTimeout will be used by the provider

"Job [OUT: CIM_ConcreteJob REF]: Job identifier

"Return Value[uint16]: Success / Failure

13.2.2.5 Release Tape

ReleaseTape marks the end of a 'transaction'. At this point, the tape becomes available for use by other clients or for a new 'transaction'. ReleaseTape shall fail if called on a tape whose activity is not "Acquired"

ReleaseTape (dest, Job)

"dest [IN:CIM_PhysicalTape REF]: Tape to release

"job [OUT: CIM_ConcreteJob REF]: Job identifier

"Return Value[uint16]: Success / Failure

13.2.2.6 PositionTape

PositionTape is used to position a tape before data gets read or written by one of the following calls: ReadTapeMetaData, WriteTapeMetaData, TapeSnapshot, and SelectiveTapeCopy. PositionTape uses relative positioning and can also be used to rewind the tape by passing 0 as a start position. PositionTape shall fail if called on a tape whose activity is not "Acquired"

PositionTape (dest, startType, start, job)

"dest [IN:CIM_PhysicalTape REF]: Destination tape

"startType [IN:uint16(enumeration)]: start position type ("filemark" or "block")

"start [IN:sint64]: start position for reading. Relative positioning implies that negative values are acceptable. A zero value has a special meaning and will trigger a full rewind of the tape.

"job [OUT: CIM_ConcreteJob REF]: Job identifier

"Return Value[uint16]: Success / Failure.

13.2.2.7 WriteTapeMetaData

WriteTapeMetaData is used to write metadata to tape at a position previously specified by a call to PositionTape. The size of the tape metadata size is limited to 1MB beyond which an error will be returned. WriteTapeMetaData shall fail if called on a tape whose activity is not "Acquired"

WriteTapeMetaData (dest, data, job)

"dest [IN:CIM_PhysicalTape REF]: Destination tape

"data[IN:TapeMetaData]: a TapeMetaData object to be written to tape

"job [OUT: CIM_ConcreteJob REF]: Job identifier

"Return Value[uint16]: Success / Failure

13.2.2.8 ReadTapeMetaData

ReadTapeMetaData is used to read metadata from tape. The tape meta data size is limited to 1MB beyond which an error will be returned.

Note that the data parameter is a reference, which means that clients will have to retrieve the actual data from the provider through CIM access methods (i.e., GetInstance). The lifecycle of tape metadata on the provider is defined as follows:

"A list of TapeMetaData instances will be maintained for every acquired tape.

"The provider will delete a given TapeMetaData instance upon receiving a GetInstance call.

"Upon receiving a ReleaseTape call, the provider will clear its list of TapeMetaData instances, thus ensuring proper memory management.

ReadTapeMetaData shall fail if called on a tape whose activity is not "Acquired"

ReadTapeMetaData (dest, sizeType, size, data, job)

"dest [IN:CIM_PhysicalTape REF]: Destination tape

"sizeType [IN:uint16(enumeration)]: type of the elements to be copied ("filemark" or "block")

"size [IN:uint32]: number of elements of type "sizeType" to be copied

"data [OUT:TapeMetaData REF]: meta data to be read.

"job [OUT: CIM_ConcreteJob REF]: Job identifier

"Return Value[uint16]: Success / Failure

13.2.2.9 TapeSnapshot

TapeSnapshot triggers a snapshot copy. It is used for simple snapshot. TapeSnapshot shall fail if called on a tape whose activity is not "Acquired"

TapeSnapshot (src, dest, copyType, job)

"src [IN:CIM_PhysicalTape REF]: Source tape

"dest [IN:CIM_PhysicalTape REF]: Destination tape

"copyType [IN:uint16(enumeration)]: Type of copy ("SimpleSnapshot")

"job [OUT: CIM_ConcreteJob REF]: Job identifier

"Return Value[uint16]: Success / Failure

13.2.2.10 SelectiveTapeCopy

SelectiveTapeCopy is used for partially copying tape data. It allows a host system to copy all or part of a tape to another tape. SelectiveTapeCopy is to be used in conjunction with WriteTapeMetaData to add metadata to tape

(See Selective Tape Copy recipe (13.3.2) for an action sequence example). SelectiveTapeCopy shall fail if called on a tape whose activity is not "Acquired"

SelectiveTapeCopy (src, dest, copyType, src, dest, sizeType, size, job)

"src [IN:CIM_PhysicalTape REF]: Source tape

"dest [IN:CIM_PhysicalTape REF]: Destination tape

"copyType=SelectiveCopy [IN:uint16(enumeration)]: type of copy

"sizeType [IN:uint16(enumeration)]: type of the elements to be copied ("filemark" or "block")

"size [IN:uint32]: number of elements of type "sizeType" to be copied

"job [OUT: CIM_ConcreteJob REF]: Job identifier

"Return Value[uint16]: Success / Failure

13.2.2.11 GetTapeCopyServicesActivity

GetTapeCopyServices indicates what copy-related actions are currently performed on a given tape. In a non-locking scenario, concurrent clients can use this call to check whether copy operations are in progress.

GetTapeCopyServicesActivity (dest, activity, job)

"dest [IN:CIM_PhysicalTape REF]: Destination tape

"activity [OUT:uint16(enumeration)]: type of copy

Activity is an enumeration type defined as follows:

"Idle: The target tape is not currently "acquired"

"Acquired: The target tape is "acquired" and no operation is currently being performed

"Writing: The target tape is "acquired" and tape meta data is being written

"Reading: The target tape is a "acquired" and meta data is being read

"Positioning: The target tape is a "acquired" and being positioned

"Copying_snapshot: The target tape is the source or destination of a snapshot copy

"Copying_selective: The target tape is the source or destination of a selective copy

"Return Value[uint16]: Success / Failure

13.2.2.12 Job Termination

These specifications do not specify means to terminate a running job. This is left up to the Job Control Profile. This has two consequences:

"Synchronous implementations of the VTL Profile methods cannot be explicitly aborted.

"Support for the Job Control Profile is conditional: if the provider features one or more asynchronous implementations of the VTL Profile methods, then it shall support the Job Control Profile.

Upon receiving a termination request for a given job, the SMI-S provider shall interrupt the specified job. These specifications do not make any recommendations as whether corrective actions should be taken It makes sense however to let the client application handle the failure, reposition the tape etc. Job termination impacts a job, not a 'transaction'. This means that another job can be started without having to reacquire the tape. As a corollary, this

also means that terminating a job doesn't preclude the client application to release the tape to mark the end the 'transaction'.

13.3 Recipes

13.3.1 Simple Snapshot recipe

The simple snapshot feature copies one piece of media to another. The source and destination may be either physical media or virtual media in a single virtual tape library system.

Using an arbitrary tape format, here's a sequence of action that would be used to perform a simple snapshot from Tape1 to Tape2:

"Obtain a lock on the destination tape: AcquireTape (Tape2, timeout, &job)

"Optionally, read meta data at the beginning of Tape2 and make sure this is the "right" tape:

- Rewind: PositionTape (Tape2, "filemark", 0)
- ReadTapeMetaData ("filemark", 1, &data, &job)
- Backup app internal validation

"Write meta data at the beginning of tape:

- Construct TapeMetaData object (data)
- Rewind: PositionTape (Tape2, "filemark", 0)
- WriteTapeMetaData(Tape2, data, &job).

"Acquire source tape: AcquireTape (Tape1, timeout, &job)

"Position tape after the first meta data section:

- Rewind: PositionTape (Tape1, "filemark", 0)
- Skip meta data: PositionTape(Tape1, "filemark", 1)

"Perform snapshot: TapeSnapshot(Tape1, Tape2, "SimpleSnapshot", &job)

"Release source and destination tapes

- ReleaseTape (Tape1, &job)
- ReleaseTape (Tape2, &job)

Note that the same result could be achieved by using the selective tape copy service passing 0 for the source start position and size arguments. To allow for vendors who do not want to support selective tape copy, I believe the simple snapshot case should remain in the specifications.

13.3.2 Selective Tape Copy recipe

Using an arbitrary tape format, here's a typical sequence of actions that could be performed to do a selective tape copy from Tape1 to Tape2 for n elements of type sizeType:

"Obtain a lock on the destination tape: AcquireTape (Tape2, timeout, &job)

"Optionally, read meta data at the beginning of Tape2 and make sure this is the "right" tape:

- Rewind: PositionTape (Tape2, "filemark", 0)

- ReadTapeMetaData (Tape2, "filemark", 1, &data, &job)1
- Backup app internal validation

"Optionally, write meta data at the beginning of tape:

- Construct TapeMetaData object (data)
- Rewind: PositionTape (Tape2, "filemark", 0)
- WriteTapeMetaData(Tape2, data, &job).

"Write meta data for this copy:

- Construct TapeMetaData object (data)
- PositionTape (Tape2, startType, destStartPosition, &job)
- WriteTapeMetaData(Tape2, data, &job)

"Acquire source tape: AcquireTape (Tape1, timeout, &job)

"Position source tape at appropriate location:

- Rewind (only if necessary - this depends on the client application): PositionTape (Tape1, "filemark", 0)
- PositionTape(Tape1, startType, srcStartPosition, &job)

"Do the copy: SelectiveTapeCopy(Tape1, handle2, "SelectiveCopy", sizeType, size, &job)

"Write some more meta data:

- Construct TapeMetaData object (data)
- WriteTapeMetaData(Tape2, data, &job).

"Possibly run other jobs…

"Release source and destination tapes

- ReleaseTape (Tape1, &job)
- ReleaseTape (Tape2, &job)

13.4 Health and Fault Management Consideration

Not supported in this version of the standard.

13.5 Cascading Considerations

None

13.6 Registered Name and Version

Tape Copy Service version 1.3.0

13.7 CIM Elements

Table 138 describes the CIM elements for Tape Copy Service.

Table 138 - CIM Elements for Tape Copy Service

Element Name	Requirement	Description
13.7.1 CIM_ElementCapabilities	Mandatory	Association linking the SNIA_TapeCopyService object to the SNIA_TapeCopyCapabilities object.
13.7.2 CIM_HostedService	Mandatory	Association linking a VLSSystem CIM_ComputerSystem object to the SNIA_TapeCopyService object.
13.7.3 SNIA_TapeCopyCapabilities	Mandatory	TapeCopyCapabilities describes functionality supported by TapeCopyService.
13.7.4 SNIA_TapeCopyService	Mandatory	Provides functions needed for 2 types of copy: snapshot copy and selective copy.
13.7.5 SNIA_TapeMetaData	Mandatory	SNIA_TapeMetaData represents backup-application-proprietary meta data that needs to be written or read to/from tape.

13.7.1 CIM_ElementCapabilities

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

Table 139 describes class CIM_ElementCapabilities.

Table 139 - SMI Referenced Properties/Methods for CIM_ElementCapabilities

Properties	Flags	Requirement	Description & Notes
ManagedElement		Mandatory	
Capabilities		Mandatory	

13.7.2 CIM_HostedService

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

Table 140 describes class CIM_HostedService.

Table 140 - SMI Referenced Properties/Methods for CIM_HostedService

Properties	Flags	Requirement	Description & Notes
Antecedent		Mandatory	
Dependent		Mandatory	

13.7.3 SNIA_TapeCopyCapabilities

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

Table 141 describes class SNIA_TapeCopyCapabilities.

Table 141 - SMI Referenced Properties/Methods for SNIA_TapeCopyCapabilities

Properties	Flags	Requirement	Description & Notes
SupportedSyncActions		Mandatory	SupportedSyncActions lists the tape copy services implemented synchronously.
SupportedAsyncActions		Mandatory	SupportedAsyncActions lists the tape copy services implemented asynchronously.
DefaultTimeout		Mandatory	DefaultTimeout is the time after which a transaction initiated by a call to TapeCopyService.AquireTape() will be released automatically by the provider if TapeCopyService.ReleaseTape() wasn't called explicitly.

13.7.4 SNIA_TapeCopyService

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

Table 142 describes class SNIA_TapeCopyService.

Table 142 - SMI Referenced Properties/Methods for SNIA_TapeCopyService

Properties	Flags	Requirement	Description & Notes
AcquireTape()		Optional	AcquireTape initializes a transaction for a given tape and informs the provider that a sequence of actions will be performed on that tape. A transaction is defined as a sequence of actions on a tape, starting with AcquireTape and ending with ReleaseTape. A call to AcquireTape is required before any sequence of actions can be performed, actions that will result in an error if the tape activity is not 'Acquired'. These actions are: PositionTape, WriteMetaData, ReadMetaData, TapeSnapshot, SelectiveTapeCopy. GetTapeServicesActivity is the only call pertaining to a tape that doesn't require that tape to be 'Acquired'. For any given tape, additional calls to AcquireTape shall return an error unless ReleaseTape was called or if the AcquireTape timeout has expired.
ReleaseTape()		Optional	ReleaseTape marks the end of a transaction. At this point, the tape becomes available for use by other clients or for a new transaction. ReleaseTape shall fail if called on a tape whose activity is not 'Acquired'.
PositionTape()		Optional	PositionTape positions a tape before data gets read or written by one of the following calls: ReadTapeMetaData, WriteTapeMetaData, TapeSnapshot, SelectiveTapeCopy. PositionTape uses relative positioning and can also be used to rewind the tape by passing 0 as a start position. PositionTape shall fail if called on a tape whose activity is not 'Acquired'.
WriteTapeMetaData()		Optional	WriteTapeMetaData writes meta data to tape at a the current tape position (specified by PositionTape). The size of the tape meta data size is limited to 1MB beyond which an error will be returned. WriteTapeMetaData shall fail if called on a tape whose activity is not 'Acquired'.
ReadTapeMetaData()		Optional	ReadTapeMetaData reads meta data from tape. The tape meta data size is limited to 1MB beyond which an error will be returned. The metaData parameter is a reference, which means that clients will have to retrieve the actual data from the provider through CIM access methods (i.e. GetInstance). The lifecycle of tape meta data on the provider is defined as follows: (1) A list of TapeMetaData instances will be maintained for every acquired tape. (2) The provider will delete a given TapeMetaData instance upon receiving a GetInstance call. (3) Upon receiving a ReleaseTape call, the provider will clear its list of TapeMetaData instances, thus ensuring proper memory management.
TapeSnapshot()		Optional	TapeSnapshot triggers a snapshot copy. TapeSnapshot shall fail if called on tapes whose activity is not 'Acquired'.

Table 142 - SMI Referenced Properties/Methods for SNIA_TapeCopyService

Properties	Flags	Requirement	Description & Notes
SelectiveTapeCopy()		Optional	SelectiveTapeCopy partially copies data. It allows a host system to copy all or part of a tape to another tape. SelectiveTapeCopy is to be used in conjunction with WriteTapeMetaData to add meta data to tape. SelectiveTapeCopy shall fail if called on a tape whose activity is not 'Acquired'.
GetTapeServicesActivity()		Optional	GetTapeCopyServices indicates what copy-related actions is currently performed on a given tape. GetTapeCopyServices shall be implemented synchronously. Concurrent clients can use this call to check whether copy operations are in progress. Tape activity is returned by way of an OUT parameter, defined as an enumeration of the following values: (1) Idle: the target is not currently acquired. (2) Acquired: the target tape is currently acquired and no operation is currently being performed. (3) Writing: the target tape is acquired and tape meta data is being written. (4) Reading: the target tape is acquired and tape meta data is being read. (5) Positioning: the target tape is acquired and being positioned. (6) SnapshotCopy: the target tape is acquired and is the source or target tape of a snapshot copy. (7) SelectiveCopy: the target tape is acquired and is the source or target tape of a selective copy.

13.7.5 SNIA_TapeMetaData

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

Table 143 describes class SNIA_TapeMetaData.

Table 143 - SMI Referenced Properties/Methods for SNIA_TapeMetaData

Properties	Flags	Requirement	Description & Notes
MetaData		Mandatory	Array of binary blocks.
PreFileMarkCount		Mandatory	Number of filemarks to be written before the meta data blocks.
PostFileMarkCount		Mandatory	Number of filemarks to be written after the meta data blocks.
BlockSizes		Mandatory	Sizes of individual binary blocks.

EXPERIMENTAL