Storage Management Technical Specification, Part 1 Overview

Version 1.6.1, 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

November 30, 2016

Revision History

Revision 1

Date

5/25/12

SCRs Incorporated and other changes

None

Revision 2

Date

27 August 2013

SCRs Incorporated and other changes

Overview part number changed to Part 1, per ISO request change re SMI-S 1.5

Comments

Editorial notes and DRAFT material are displayed.

Revision 3

Date

4 December 2013

SCRs Incorporated and other changes

None

Comments

Editorial notes are displayed. DRAFT material is hidden.

Revision 4

Date

25 February 2014

SCRs Incorporated and other changes

None

Comments

Editorial notes and DRAFT material are hidden.

Revision 5

Date

11 August 2014

SCRs Incorporated and other changes

None

Comments

Editorial notes and DRAFT material are hidden.

Revision 6

Date
11 October 2016

SCRs Incorporated and other changes
None

Comments
Editorial notes and DRAFT material are hidden.

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/.

USAGE

The SNIA hereby grants permission for individuals to use this document for personal use only, and for corporations and other business entities to use this document for internal use only (including internal copying, distribution, and display) provided that:

1) Any text, diagram, chart, table or definition reproduced shall be reproduced in its entirety with no alteration, and,

2) Any document, printed or electronic, in which material from this document (or any portion hereof) is reproduced shall acknowledge the SNIA copyright on that material, and shall credit the SNIA for granting permission for its reuse.

Other than as explicitly provided above, you may not make any commercial use of this document, sell any or this entire document, or distribute this document to third parties. All rights not explicitly granted are expressly reserved to SNIA.

Permission to use this document for purposes other than those enumerated above may be requested by e-mailing tcmd@snia.org. Please include the identity of the requesting individual and/or company and a brief description of the purpose, nature, and scope of the requested use.

All code fragments, scripts, data tables, and sample code in this SNIA document are made available under the following license:

BSD 3-Clause Software License

Copyright (c) 2014-2016, The Storage Networking Industry Association.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

- Neither the name of The Storage Networking Industry Association (SNIA) nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

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.

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.

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

Maturity Level

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

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.
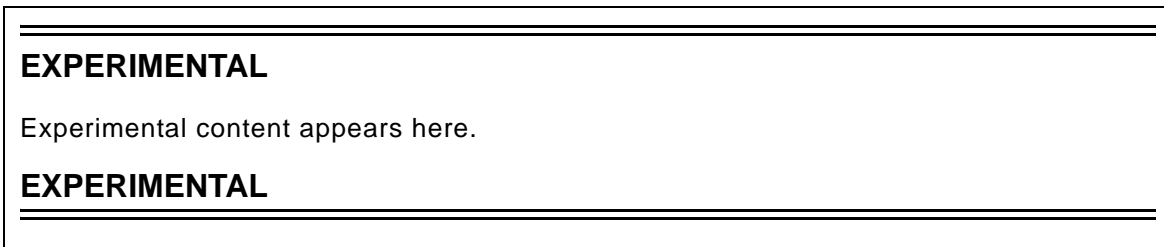
```
EXPERIMENTAL

Experimental content appears here.

EXPERIMENTAL
```

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.
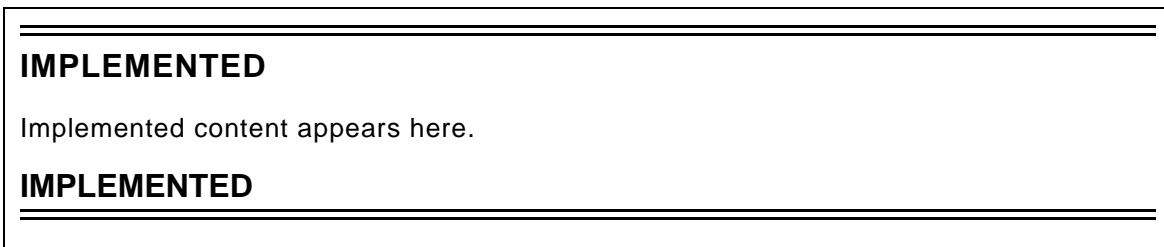
```
IMPLEMENTED

Implemented content appears here.

IMPLEMENTED
```

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.
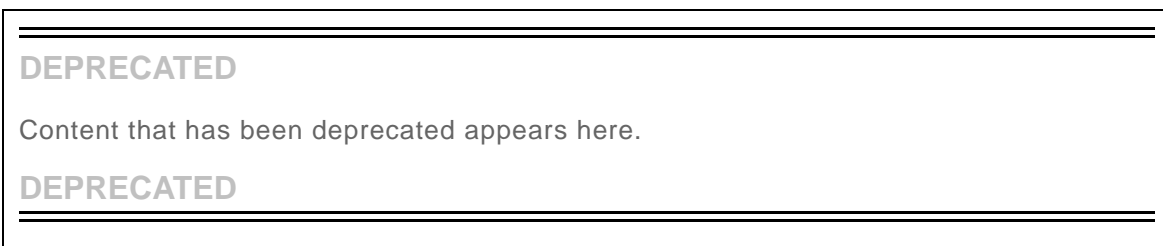
```
┌─────────────────────────────────────────────────────────────────┐
│ ═══════════════════════════════════════════════════════════════ │
│                                                                   │
│ DEPRECATED                                                        │
│                                                                   │
│ Content that has been deprecated appears here.                    │
│                                                                   │
│ DEPRECATED                                                        │
│ ═══════════════════════════════════════════════════════════════ │
└─────────────────────────────────────────────────────────────────┘
```

Figure 4 - Deprecated Tag

Contents

List of Figures

List of Tables

Foreword

The Overview part of the Storage Management Technical Specification contains informative clauses that provide an overview of how SMI-S works. It is a useful base for understanding the details of the technical specification. While the normative information of the specification is contained in other parts, this part provides high-level introductory material on key concepts of the specification.

Parts of this Standard

This standard is subdivided in the following parts:

- *Storage Management Technical Specification, Part 1 Overview, 1.6.1 Rev 6*

- *Storage Management Technical Specification, Part 2 Common Architecture, 1.6.1 Rev 6*

- *Storage Management Technical Specification, Part 3 Common Profiles, 1.6.1 Rev 6*

- *Storage Management Technical Specification, Part 4 Block Devices, 1.6.1 Rev 6*

- *Storage Management Technical Specification, Part 5 Filesystems, 1.6.1 Rev 6*

- *Storage Management Technical Specification, Part 6 Fabric, 1.6.1 Rev 6*

- *Storage Management Technical Specification, Part 7 Host Elements, 1.6.1 Rev 6*

- *Storage Management Technical Specification, Part 8 Media Libraries, 1.6.1 Rev 6*

SNIA Web Site

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

SNIA Address

Requests for interpretation, suggestions for improvement and addenda, or defect reports are welcome. They should be sent via the SNIA Feedback Portal at http://www.snia.org/feedback/ or by mail to the Storage Networking Industry Association, 4360 ArrowsWest Drive, Colorado Springs, Colorado 80907, U.S.A.

1 Scope

This Technical Specification defines an interface for the secure, extensible, and interoperable management of a distributed and heterogeneous storage system. This interface uses an object-oriented, XML-based, messaging-based protocol designed to support the specific requirements of managing devices and subsystems in this storage environment. Using this protocol, this Technical Specification describes the information available to a WBEM Client from an SMI-S compliant CIM WBEM Server.

2 Normative References

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

2.1   Approved references

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

2.2   References under development

Storage Management Technical Specification, Part 2 Common Architecture, 1.6.1 Rev 6

Storage Management Technical Specification, Part 3 Common Profiles, 1.6.1 Rev 6

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

2.3   Other references

DMTF DSP0214:2004 CIM Operations over HTTP

3 Definitions, Symbols, Abbreviations, and Conventions

For the purposes of this document, the definitions, symbols, abbreviations, and conventions given in *Storage Management Technical Specification, Part 2 Common Architecture, 1.6.1 Rev 6* apply.

4 Introduction

4.1 Preamble

Large Storage Systems and Storage Area Networks (SANs) are emerging as a prominent and independent layer of IT infrastructure in enterprise class and midrange computing environments. Examples of applications and functions driving the emergence of new storage technology include:

- Sharing of vast storage resources between multiple systems via networks,

- LAN free backup,

- Remote, disaster tolerant, on-line mirroring of mission critical data,

- Clustering of fault tolerant applications and related systems around a single copy of data.

- Archiving requirements for sensitive business information.

- Distributed database and file systems.

To accelerate the emergence of more functional and sophisticated storage systems in the market, the industry requires a standard management interface that allows different classes of hardware and software products supplied by multiple vendors to reliably and seamlessly interoperate for the purpose of monitoring and controlling resources. The SNIA Storage Management Initiative (SMI) was created to develop this specification (SMI-Specification or SMI-S), the definition of that interface. This standard provides for heterogeneous, functionally rich, reliable, and secure monitoring/control of mission critical global resources in complex and potentially broadly-distributed, multi-vendor storage topologies like SANs. As such, this interface overcomes the deficiencies associated with legacy management systems that deter customer uptake of more advanced storage management systems.

4.2 Business Rationale

This interface is targeted at creating broad multi-vendor management interoperability and thus increasing customer satisfaction. To that end, this specification defines an "open" and extensible interface that allows subsystems and devices within the global context of a large storage system to be reliably and securely managed by overlying presentation frameworks and management systems in the context of the rapidly evolving multi-vendor market. In specific, SAN integrators (like end-users, VARs, and SSPs) can, via this standardized management interface, more flexibly select between multiple vendors when building the hierarchy of software systems required to manage a large storage system independent of the underlying hardware systems. Additionally, storage integrators can more flexibly select between alternate hardware vendors when constructing storage configurations. Broad adoption of the standards defined and extended in this specification will provide increased customer satisfaction and will:

- More rapidly expand the acceptance of new storage management technology like SANs and iSCSI;

- Accelerate customer acquisition of new storage management technology;

- Expand the total market.

Additionally, a single common management interface allows SAN vendors and integrators to decrease the time required to bring new more functional technology, products, and solutions to market.

4.3   Interface Definition

This management interface allows storage management systems to reliably identify, classify, monitor, and control physical and logical resources in a storage system. The fundamental relationship of this interface to storage management software, presentation frameworks, user applications, SAN physical entities (i.e., devices), SAN discovery systems, and SAN logical entities is illustrated in Figure 5.
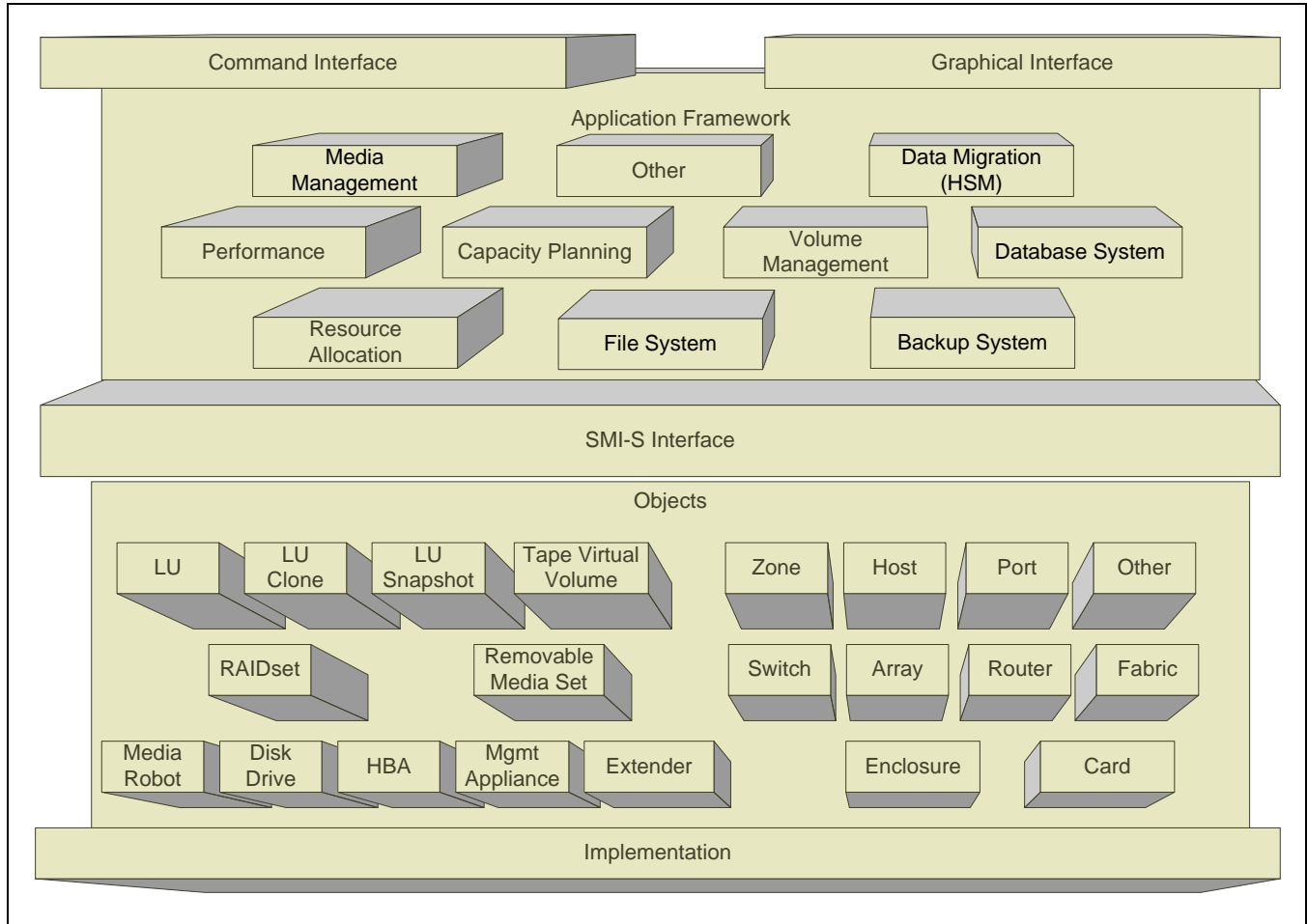

Figure 5 - Interface Functions

Figure 5 illustrates that functions of the interface can be distributed across multiple devices (i.e., Switches or Array Controllers) and/or software systems (i.e., Discovery Systems). While the functionality of the interface is distributed within or across a storage environment, to insure that monitoring and control operations by clients are consistent and reliable, the state of a given resource is not certain to be valid if it is simultaneously available to clients from multiple unsynchronized sources.

> **EXAMPLE:** A request by an SRM application and a backup engine for the bandwidth available on a given Fibre Channel path should be coordinated by a single monitoring entity to insure information consistency. If the SRM application and Backup engine obtain different available bandwidth information for a given Fibre Channel path from multiple unsynchronized sources they could function in conflict and degrade the efficiency of the environment.

Addressing this concern is the responsibility of parties configuring Storage and Network management clients that rely on the primitives defined in the specification.

NOTE   Within this architecture (as depicted by Figure 5) entities like an appliance-based volume manager may potentially act as both a client and a server to the interface.

> **EXAMPLE:** A Host-based volume manager wants to construct a large storage pool from multiple SAN appliance based volumes, as well as volumes/LUNs originating from array controllers. In this case, the host based volume manager needs to inspect the characteristics of the volumes on both the SAN appliance and array controller prior to allocation. Additionally, the SAN appliance (which runs a volume manager) needs to inspect the properties of storage devices when building its volumes. As such, the SAN appliance in this case is both a client and server in the management environment, depending on the action being performed.

Figure 5 includes a number of strategic functional requirements for the interface. These capabilities will be introduced to the interface implementation over time, and may not be present in this version of the interface. The functionalities required to fully satisfy the needs of clients using a storage management interface include:

a) Clients need to be able to obtain sufficient information to discern the topology of the SAN or complex storage system;

b) Clients need to be able to reliably identify resources that have experienced an error/fault condition that has resulted in degraded/disabled operation;

c) Clients need to be able to construct a zone of allocation around a select group of host and storage resources;

d) Clients need to be able to identify nonvolatile storage resources available to a storage management system, to allow them to construct a storage pool of a consistent level of performance and availability;

e) Clients need to be able to identify third-party copy engines (and associated media libraries/robots) available to a cooperating backup engine, allowing it to allocate an engine/library/robot to a given backup task;

f) Clients need to be able to dynamically allocate non-volatile storage resources;

g) Each volume to be utilized is subject to strict availability and performance requirements. As a result, the file system needs to inspect the properties of each volume prior to allocation.

h) Clients need to be able to access sufficient topology and component information to allow a Storage Resource Management (SRM) application like a performance monitor to examine topology and line utilization, such that performance bottlenecks can be exposed and capacity planning performed;

i) Clients need to be able to employ appropriate data reporting and tracking to allow capacity planning system to identify each storage pool in the SAN and then interact with the manager of each pool to assess utilization statistics;

j) Clients need to be provided with adequate controls for a privileged, user-written application to restrict the use of a volume to a specific host, set of hosts, or set of controller communications ports;

k) Clients need to be assured of timely propagation of data concerning the health and performance of the devices and subsystems in the SAN to fault isolation and analysis systems.

Example non-goals for this interface include:

a) The ability to select a logical communications port over which to send/receive data;

b) The ability to read or write data to a volume;

c) The ability to identify and recover from data communications errors and failures;

d) The ability to log a new communications device into a network.

4.4   Technology Trends

To be broadly embraced and long lived this management interface should respect and leverage key technology trends evolving within the industry. These include:

a) *Improved Connectivity*: Whether available In-band (i.e., over Fibre Channel/iSCSI) or available out-of-band (i.e., over a LAN/MAN/WAN), or available over a mix of both, virtually all devices in a storage management environment have (or soon will have), access to a common communications transport suitable for carrying management information content (e.g., TCP/IP), that is used to transmit a standardized encoding (e.g., a WBEM Protocol) of recognized semantics (e.g., CIM);

b) *Increased Device Manageability*: Through a common, general-purpose network transport, provide the option to provide proxy services to provide access to (e.g., general purpose computer system) devices via this standardized management interface;

> **EXAMPLE:** A legacy array controller is incapable of running the software necessary to implement a management server for this interface and uses a proxy server on a SAN appliance to communicate within the management environment.

> **EXAMPLE:** An HBA is incapable of running the software necessary to implement a management server for this interface and uses a proxy server on its host system to communicate within the management environment.

c) *XML Standardization:* XML is providing the ability to create management protocols with an extensible, platform independent, human readable, content describable communication language. This streamlines the task of developing infrastructure to support his interface and debug systems around the interface.

d) *Object Independent Protocols*: These protocols provide appropriate abstraction – separating the definition of the object model from the semantics/syntax of the protocol. Additionally, the transport-independent, content-description (i.e., markup) nature of XML allows it to be utilized by both web-enabled application and appliances;

e) *Increased SAN Complexity*: SANs are being configured with diverse classes of components and widely distributed topologies. Management clients and servers in the environment need to anticipate being widely distributed on systems, appliances and devices throughout large SAN topologies, while maintaining real-time distributed state for logical entities. Figure 6 provides an example of a single SAN built from multiple classes of components spanning three physical locations (i.e., Sites A, B and C).
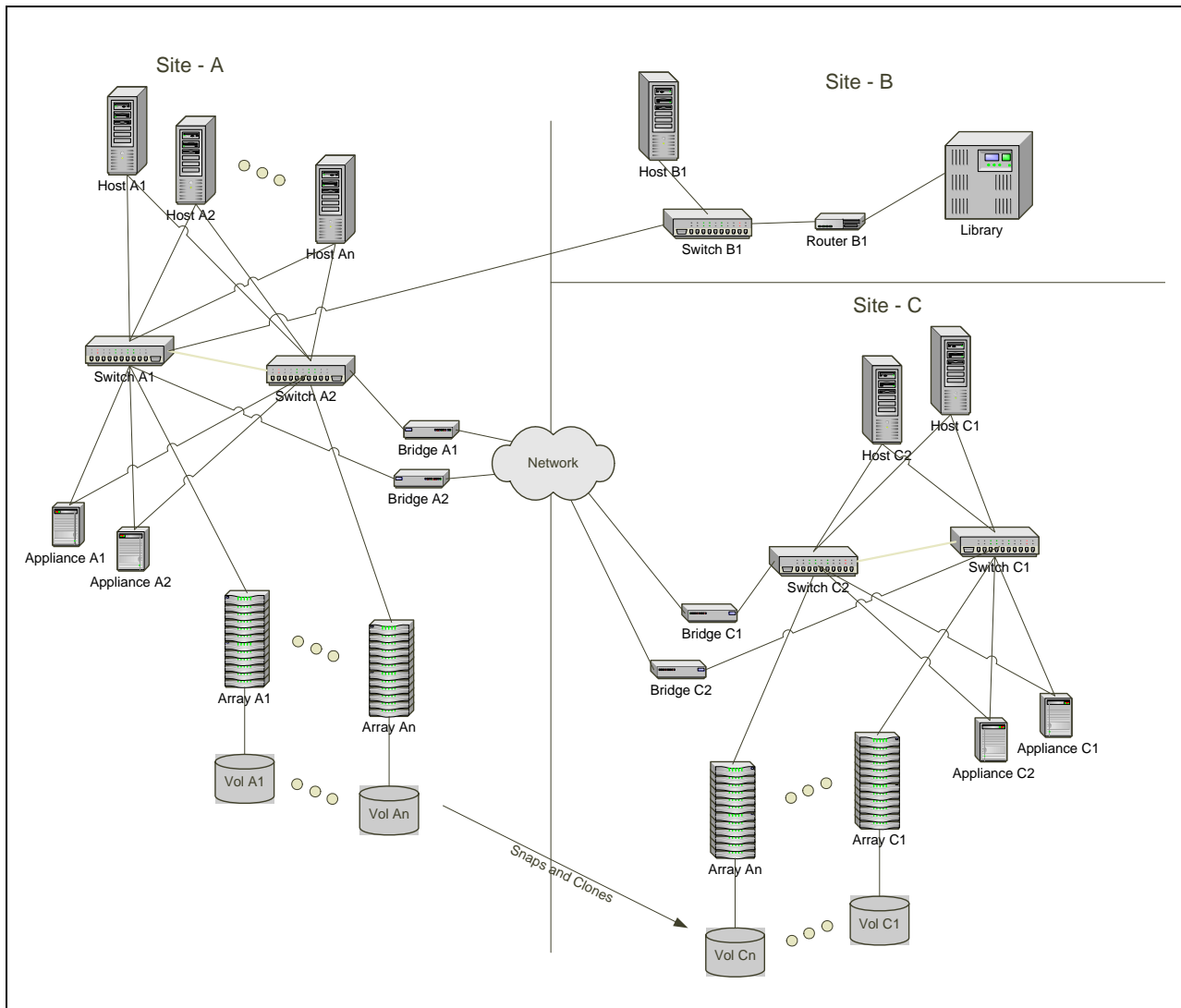
Figure 6 - Large SAN Topology

4.5   Management Environment

Clients and Servers of this interface can be widely distributed on systems, appliances, and devices across a network that includes one or more large SAN topologies.

The configuration in Figure 7 provides an example client/server distribution using in-band TCP/IP communications, out of band TCP/IP communications, or employing proxy services to bridge legacy and/ or proprietary communication interfaces. The device "Old Array Controller" is incapable of appropriate communication with clients and servers in the management environment to provide management access (i.e., a CIM Server). Access to the communications transport that clients and servers share for communication is achieved via a proxy service on the host computer in the upper right hand corner of Figure 7. All other clients and servers communicate via direct access to a common communications transport.
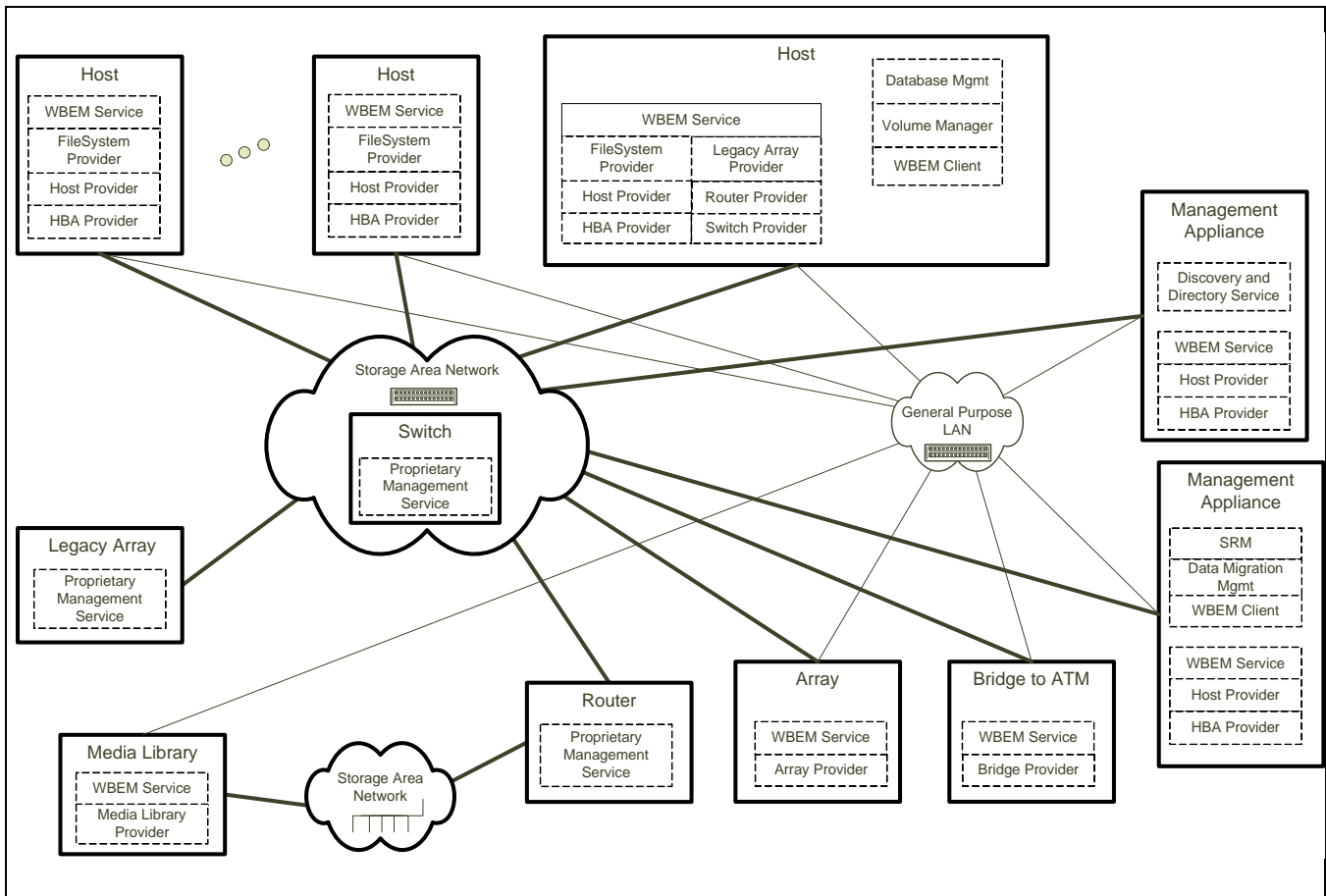
Figure 7 - Example Client Server Distribution in a SAN

4.6   Architectural Objectives

The following reflect architectural objectives of the interface. Some of these capabilities are not present in the initial release of the interface, but are inherent in its architecture and intended extensibility. They are intended to provide guidance concerning the present and future direction of development of the *Storage Management Technical Specification*.

a) Consistency: State within a managed object and between objects remains consistent independent of the number of clients simultaneously exerting control, the distribution of objects in the environment, or the management action being performed;

b) Isolation: A client that needs to execute an atomic set of management actions against one or more managed objects is able to do so in isolation of other clients, who are simultaneously executing management actions against those same objects;

c) Durability: Consistency, and isolation are preserved independent of the failure of any entity or communications path in the management environment;

d) Consistent Name Space: Managed objects in a single management domain adhere to a consistent naming convention independent of state or reliability of any object, device, or subsystem in the SAN;

e) Distributed Security: Monitoring and control operations are secure. The architecture supports:

1) Client authentication;

2) Privacy (encryption) of the content of the messages in this protocol;

3) Client authorization;

f) Physical Interconnect Independence: The interface will function independent of any particular physical interconnect between components, any supplier, or any topology;

g) Multi-vendor Interoperability: Clients and servers should use a common communication transport and message/transfer syntax to promote seamless plug compatibility between heterogeneous multi-vendor components that implement the interface;

h) Scalability: The size, physical distribution, or heterogeneity of the storage system does not degrade the quality or function of the management interface;

i) Vendor Unique Extension: The interface allows vendors to implement proprietary functionality above and beyond the definitions here-in to distinguish their products and services in the market independent of the release of a new version of the interface;

j) Volatility of State: This interface does not assume that objects are preserved in non-volatile repositories. Clients and servers may preserve object state across failures, but object preservation is not mandatory;

k) Replication: This interface provides no support for the automatic replication of object state within the management environment;

l) Functional Layering Independence: The design of this interface is independent of any functional layering a vendor chooses to employ in constructing the storage management systems (hardware and software) necessary to manage a storage environment;

m) Asynchronous or Synchronous execution: Management actions may execute either asynchronously or synchronously;

n) Events: This interface provides for the reliable asynchronous delivery of events to one or more registered clients;

o) Cancelable Management Actions: Long running synchronous or asynchronous directives need to be capable of being cancelled by the client. Cancellation needs to result in the termination of work by the server and resource consumed being released;

p) Durable Reference: Object classes that persist across power cycles and need to be monitored and controlled independent of SAN reconfiguration (i.e., logical volumes) need be identified via "Durable Names" to insure consistent reference by clients;

q) Dynamic installation and reconfiguration: New clients and servers need to be capable of being added to or removed from an SMI-S management environment without disrupting the operation of other clients or servers. In most cases, clients should be capable of dynamically managing new servers that have been added to an SMI-S environment.

r) Automatic discovery of new servers: When new management servers are added to the management system they should automatically become available to management clients without the need for manual configuration by administrations staff.

4.7   Disclaimer

The SNIA makes no assurance or warranty about the interoperability, data integrity, reliability, or performance of products that implement this specification.

5 Overview

5.1 Base Capabilities

To achieve the architectural objectives and support the key technological trends in 4, "Introduction", this document describes an extensible, secure, auto-discoverable, object-oriented, XML-based messaging based interface designed to support the specific requirements of managing devices in and through storage systems. The top level protocol that implements this messaging based interface in this revision of the specification is called Web Based Enterprise Management (WBEM) and more specifically CIM/XML over http. To quickly become ubiquitous, SMI-S seeks to the greatest extent possible to leverage a number of existing enterprise management standards through this interface, such as:

- The Distributed Management Task Force (DMTF) authored Common Information Model (CIM) and Web Based Enterprise Management (WBEM) standards,

- The standards written by ANSI on Fibre Channel and SCSI,

- The World Wide Web Consortium (W3C) for standards on XML,

- The Internet Engineering Task Force (IETF) for standards on HTTP, SLP, and iSCSI.

5.2 Object Oriented

A hierarchy of object classes with properties (a.k.a. attributes) and methods (a.k.a. directives) linked via the Universal Modeling Language (UML) modeling constructs of inheritance and associations defines most of the capabilities of SMI-S. The SMI-S object model (which constitutes the bulk of this specification) is integrated with and part of the Common Information Model (CIM) at the DMTF. Implementers of this specification are encouraged to consult one of the many publicly available texts on UML or the uml.org web site (www.uml.org) to develop an understanding of UML. A brief tutorial on UML is provided in the introduction material *Storage Management Technical Specification, Part 2 Common Architecture, 1.6.1 Rev 6:* 6 Object Model General Information in this specification.
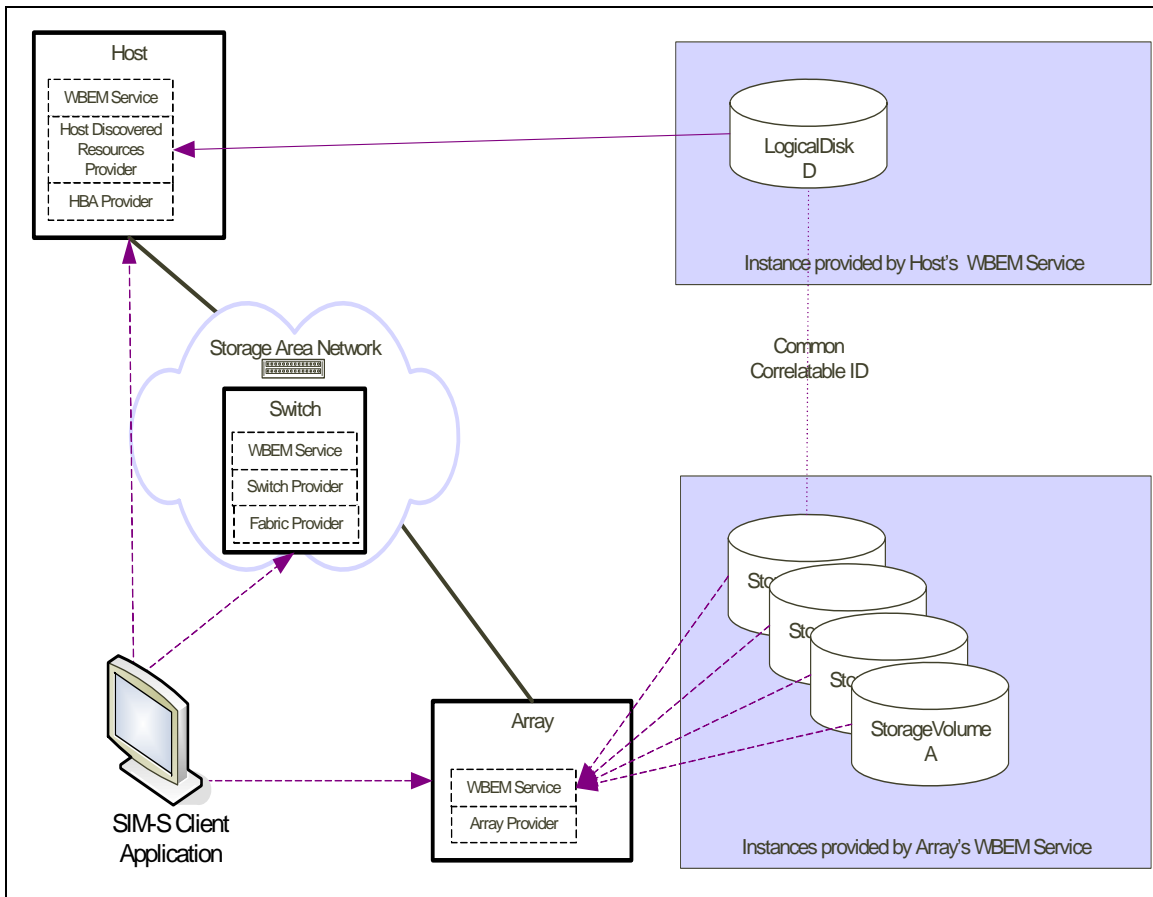
Figure 8 - Object Model/Server Relationship

In Figure 8, an SMI-S client obtains object classes and instances that it can use to manage the storage. At this level of discussion, the focus is on SMI-S conformant WBEM Clients and Servers. The WBEM Servers have providers for the various components that are responsible for the class and association instances that allow the underlying component implementation to be managed.

A standard, object-oriented interface, together with a standard interface protocol, allows WBEM Clients to discover, monitor, and control storage and network devices, regardless of the underlying implementation of those devices.

The goal of this document is to clearly and precisely describe the information expected to be available to a WBEM Client from an SMI-S compliant WBEM Service. It relies upon UML diagrams, easy-to-use tables and machine-readable, CIM-compliant Managed Object Format (MOF) (through the CIM model maintained at the DMTF). This is intended to ease the task of client implementation and to ease the task of using existing WBEM Servers. It should be noted that the MOF Interface Description Language is a precise representation of the object model in this specification, and developers are encouraged to learn this means of expression when implementing this interface. Programmers implementing this interface should reference MOF representations of the object model when faced with implementation decisions.

SMI-S compliant WBEM Servers provide instances in a manner conformant to one or more SMI-S profiles (*Storage Management Technical Specification, Part 3 Common Profiles, 1.6.1 Rev 6:* 4 Profile Introduction). The object model supporting these instances may be extended by the vendor as long as it remains conformant to the relevant SMI-S profiles. Generally, vendor-unique code is necessary in a WBEM client to take

advantage of vendor defined model extensions. Regardless of the presence of vendor extensions, a generic WBEM client is able to leverage all SMI-S features defined for a supported profile.
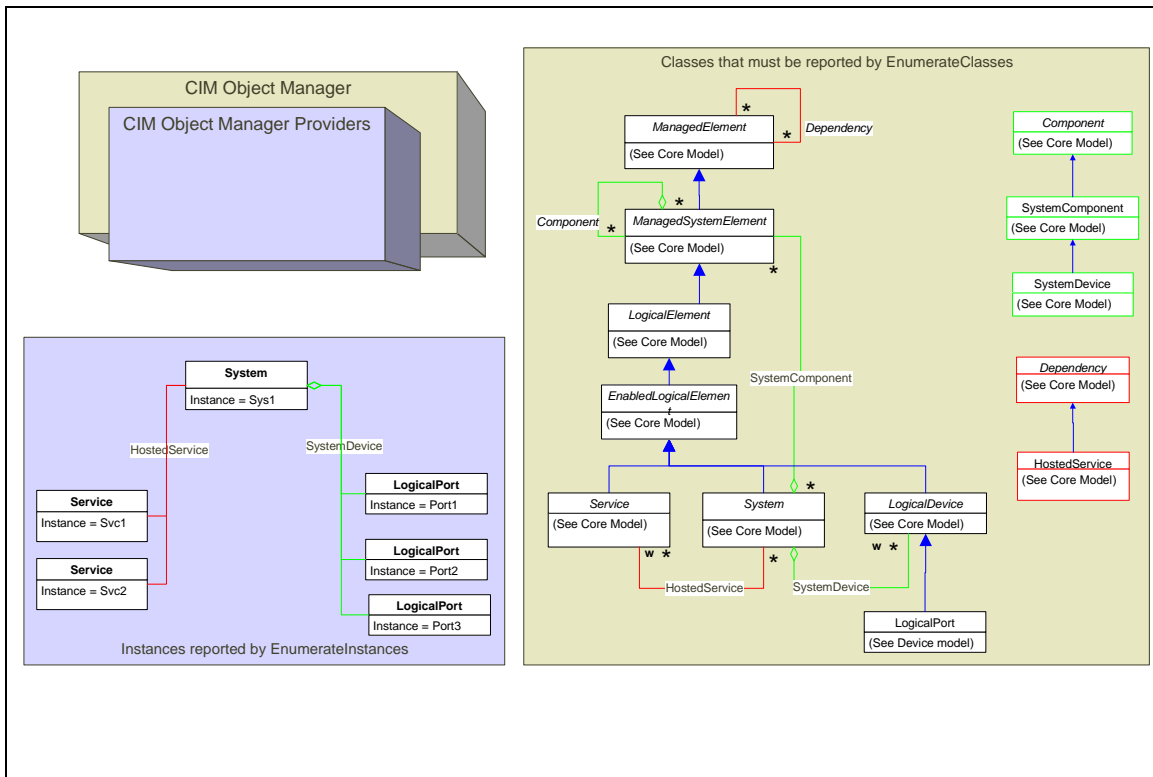
Figure 9 illustrates this requirement.

Figure 9 - Canonical Inheritance

Figure 9 illustrates that even though a Fibre Channel Switch may only report instances and allow associated method execution for certain objects, when asked by a client to enumerate its Object Classes it reports the entire hierarchy of classes in its tree. Similarly a server that instantiates an array controller reports the complete set of object classes that links it to the base canonical object of the SMI-S model. It is this single canonical root that allows any SMI-S client to discover, map, and operate upon the complete set of objects in a given SAN.

The object model presented in this specification is intended to facilitate interoperability not limiting the expression of unique features that differentiate manufacturers in the market For this reason, the object model provided only serves as a"core" to compel multi-vendor interoperability. In the interest of gaining a competitive advantage, a given vendor's implementation of the interface may include additional object classes, properties, methods, events, and associations around this "core". These vendor-unique extensions to the object model may, in select cases (e.g., extrinsic methods), require the modification of client code above and beyond that required to support the core.

6 Functionality Matrix

6.1   Overview

The functionality enabled by this version of the *Storage Management Technical Specification* follows a multi-level model. Within each level of this model, several broad categories of management are described. This creates a functionality matrix, which serves two purposes. First, it organizes a complex set of capabilities enabled by the overall SMI-S approach. Second, it helps to ensure good management functionality coverage for the managed devices comprehended by SMI-S. This section provides an overview of the functionality matrix approach for describing the management functionality provided by this version of SMI-S. A blank functionality matrix is provided in Table 1.

Table 1 - Functionality Matrix

	Fault Management	Configuration Management	Accounting Management	Performance Management	Security Management
Application Level					
File / Record Level					
Block Level					
Connectivity Level					
Device Level					

6.2   Multi-Level Model Of Networked Storage Management Functionality

The lowest level of the multi-level model of networked storage management functionality applies to managing the basic physical aspects of the elements found in a networked storage environment, and the upper levels are involved with managing the different logical levels supported by these managed elements. Each level in this model depends upon the lower levels being in place.

Shown in top-down order, the functionality levels are:

- (Level 5) Application Level Functionality,

- (Level 4) File/Record Level Functionality,

- (Level 3) Block Level Functionality,

- (Level 2) Connectivity Level Functionality,

- (Level 1) Device Level Functionality.

Managed physical elements in a networked storage environment shall support Level 1 functionality, and may support additional functionality levels as well, depending upon the logical capabilities of the managed physical element. The functionality supported by a managed element will normally involve a contiguous set of levels in this model. If a managed physical element supports functionality for a particular upper level, then it will also support functionality for each level between that level and Level 1.

As an example of this last point, consider a NAS Head device. It has a physical component (Level 1). It is connected to other physical components in the networked storage environment (Level 2). It deals with Block storage (Level 3), and it deals with Files (Level 4). A NAS Head device can therefore be expected to support functionality in levels 1 through 4 of this multi-level model of networked storage management functionality. Similarly, a regular NAS device would support management functionality in each of these same levels, although the functionality supported within each level might be slightly different, since the regular NAS device does not have a SAN back-end.

6.3 FCAPS

Within each level of this model, a basic set of functionality is needed that allows management applications to exercise FCAPS capabilities over elements supporting that level. FCAPS is a model of the working objectives of network management, and these same concepts are applied to each of the levels in the multi-level model of networked storage management functionality. A summary of FCAPS capabilities includes:

- **F**ault Management: Identifying, isolating, correcting, and logging managed element faults. Includes running diagnostics, generating fault alarms, and keeping error statistics,

- **C**onfiguration Management: Discovering, configuring, and monitoring managed elements. Includes adding, altering, and removing managed elements,

- **A**ccounting Management: Measuring and tracking usage of managed elements or services. Includes distributing resources, setting quotas, and billing,

- **P**erformance Management: Monitoring of performance, error rate, and utilization metrics for managed elements. Includes setting thresholds, problem reporting, logging of data, and examining historical data

- **S**ecurity Management: Ensuring legitimate use of managed elements or services. Includes checking user access rights, maintaining an audit trail log, generating security events and alarms, and maintaining data confidentiality where necessary.

By specifying FCAPS capabilities within each of its levels, this multi-level model is used to describe the functionality that is provided by SMI-S overall, and by individual profiles and subprofiles. The actual degree of support for FCAPS capabilities within each level is determined by individual SMI-S profiles.

6.4 Management Functionality Within Each Level Of The Model

6.4.1 (Level 1) Device Level Functionality

This level includes all functionality needed to allow management applications to deal with the physical aspects of managed elements in the networked storage environment. The physical aspects of HBAs, Switches, Storage Systems etc. are handled by functionality in this level. This level also handles functionality that is not exposed to other elements in the networked storage environment, like the managing of storage devices within a Storage System prior to their being allocated to storage pools that are accessible over the data network.

6.4.2 (Level 2) Connectivity Level Functionality

This level includes all functionality associated with allowing management applications to deal with the logical aspects of the managed connectivity between physical elements in the networked storage environment. This level is where things like Fibre Channel Fabrics and Zones are handled, and is also where iSCSI Sessions are handled. This level also handles the logical aspects of switch and extender connectivity.

6.4.3 (Level 3) Block Level Functionality

This level includes all functionality necessary to allow management applications to deal with storage volumes in a networked storage environment. This level applies to Logical Units, LUN Masking and Mapping, block aggregators like Volume Managers, etc. It also applies to block-level virtualization.

6.4.4 (Level 4) File/Record Level Functionality

This level includes all functionality associated with allowing management applications to deal with data objects like file systems in a networked storage environment. Note that this level not only applies to file systems -- it is also applies to records, for the structured usage of block storage by middleware applications such as databases and e-mail servers. This level provides the functionality that enables

management applications to determine the capacity utilization of the storage volumes handled by the Block Level Functionality.

6.4.5 (Level 5) Application Level Functionality

This level includes all functionality needed to allow management applications to deal with managed applications in the networked storage environment. This level applies to database applications, e-mail server applications, etc. that work directly with the data objects handled by the File/Record Level Functionality.

6.5 Referring To Levels And Capabilities In The Multi-level Model

To simplify talking about the different levels and capabilities within this multi-level model of networked storage management functionality, the following short-hand notation may be used in SMI-S.

Individual functionality levels are referred to as L1 through L5, and a single letter appended to this level indicates a particular kind of FCAPS capability. For instance, fault management functionality within the connectivity layer would be referred to as L2F functionality, and configuration management functionality for a physical device would be referred to as L1C functionality.

6.6 Functionality Descriptions in SMI-S Profiles

To make it easier to understand the management functionality coverage provided by individual profiles and subprofiles in this SMI-S document, each profile lists the functionality provided by the profile and its subprofiles. If a function is provided by a subprofile, this is indicated, including whether the subprofile is optional or required. Functionality listed in the profile is organized by Level, and within each Level by FCAPS category, as defined here by the Functionality Matrix.

6.7 Capabilities of This Version

This section summarizes, at a high level, the capabilities provided by this SMI-S version based on the Functionality Matrix, and is organized by Level.

6.7.1 Device Level

6.7.1.1 Fault Management

SMI-S device profiles that include the Health Package (*Storage Management Technical Specification, Part 3 Common Profiles, 1.6.1 Rev 6:* 25 Health Package) provide capabilities for reporting of the SAN device health and status, including the type, category, and source of the failures. Asynchronous notification for changes in device health status is also provided via the Indications Subprofile (*Storage Management Technical Specification, Part 3 Common Profiles, 1.6.1 Rev 6:* 42 Indication Profile).

6.7.1.2 Configuration Management

SMI-S defines the capabilities needed for the discovery, configuration, and monitoring of devices in a SAN. Asynchronous notification for changes in device configuration is provided via the Indications Subprofile (see *Storage Management Technical Specification, Part 3 Common Profiles, 1.6.1 Rev 6:* 42 Indication Profile).

6.7.1.3 Accounting Management

Other than basic device discovery, SMI-S provides no specific capabilities for device Accounting Management.

6.7.1.4 Performance Management

SMI-S enables performance management of some SAN devices (see *Storage Management Technical Specification, Part 4 Block Devices, 1.6.1 Rev 6:* 7 Block Server Performance Subprofile).

6.7.1.5   Security Management

SMI-S provides device-level security via basic authentication capabilities. See the SMI-S Security section (*Storage Management Technical Specification, Part 2 Common Architecture, 1.6.1 Rev 6:* 13 Security) and Device Credentials Profile (*Storage Management Technical Specification, Part 3 Common Profiles, 1.6.1 Rev 6:* 46 Device Credentials Subprofile) for more information. Note that the secure communication between a device proxy CIM Server and the device is outside of the scope of SMI-S.

6.7.2   Connectivity Level

6.7.2.1   Fault Management

SMI-S provides the capability to identify the heath of interconnects between SAN devices, mainly via LogicalPort.OperationalStatus (see all the port subprofiles, and the Switch Profile). Asynchronous notification for changes in link health status is also provided via the Indications Subprofile (*Storage Management Technical Specification, Part 3 Common Profiles, 1.6.1 Rev 6:* 42 Indication Profile).

6.7.2.2   Configuration Management

SMI-S defines the capabilities needed for the discovery, configuration, and monitoring of interconnects between devices in a SAN. Asynchronous notification for changes in the fabric configuration is provided via the Indications Subprofile.

6.7.2.3   Accounting Management

Connectivity-level Accounting Management is enabled in SMI-S via basic discovery capabilities and usage tracking via the optional Fabric Path Performance Subprofile.

6.7.2.4   Performance Management

SMI-S enables performance management of SAN Interconnects via both the FCPortStatistics (*Storage Management Technical Specification, Part 6 Fabric, 1.6.1 Rev 6:* 12.8.14, "CIM_FCPortSettings") class and also the transport-independent Fabric Path Performance Subprofile.

6.7.2.5   Security Management

SMI-S provides Connectivity-level security via basic device authentication capabilities, Zone Control and Enhanced Zoning Subprofiles, and the Fabric Security subprofile.

6.7.3   Block Level

6.7.3.1   Fault Management

SMI-S Block-level profiles that include the Health Package (*Storage Management Technical Specification, Part 3 Common Profiles, 1.6.1 Rev 6:* 25 Health Package) provide capabilities for reporting of block level health and status, including the type, category, and source of the failures.

6.7.3.2   Configuration Management

SMI-S defines the capabilities needed for the discovery, configuration, and monitoring block-level resources. This includes ability to discover, create, delete, and modify StorageVolumes in the SAN.

6.7.3.3   Accounting Management

SMI-S enables accounting management of Block-level resources via basic discovery and discovery of access rights and mappings.

6.7.3.4   Performance Management

SMI-S provides performance management capabilities for SAN Block-level resources (as provided by Arrays, Virtualization systems and Volume Managers) via the Block Server Performance Subprofile (*Storage Management Technical Specification, Part 4 Block Devices, 1.6.1 Rev 6:* 7 Block Server Performance Subprofile).

6.7.3.5   Security Management

SMI-S provides the ability to manage (create/delete, enable/disable) connectivity and access rights to Storage Volumes in the SAN.

6.7.4   File/Record Level

6.7.4.1   Fault Management

SMI-S NAS profiles provide Indications support on OperationalStatus for the FileSystems and FileShares.

6.7.4.2   Configuration Management

SMI-S NAS profiles provide discovery of logical storage (StoragePools) and storage extents on logical disks.

6.7.4.3   Accounting Management

This version of SMI-S defines no unique accounting management capabilities at the File level.

6.7.4.4   Performance Management

This version of SMI-S defines no unique performance management capabilities at the File level.

6.7.4.5   Security Management

This version of SMI-S defines no unique security management capabilities at the File level.

6.7.5   Application Level

This version of SMI-S does not address functionality at the application level.

7 Operational Environment

7.1 General

Figure 10 illustrates activities that either clients or servers need to account for in or to provide facilities to support:

- The discovery of constituents in the managed environment;

- The discovery of object classes as well as related associations, properties, methods, indications, and return status codes that are provided by servers in the managed environment;

- The security or resources and communications in the environment;

- The locking of resources in the presence of non-cooperating clients; (the definition of locking is left for a future version of the specification)

- The marshalling/un-marshalling of communication messages;

- The execution of basic methods that are "intrinsic" to the construction, traversal, and management of the object model provided by the distributed servers in a SAN;

- The execution of object specific "extrinsic" methods that provide clients the ability to change the state of entities in the SAN.

In addition, to facilitate ease of installation, startup, expansion, and upgrade requirements for implementations are specified for the developers of clients and servers.
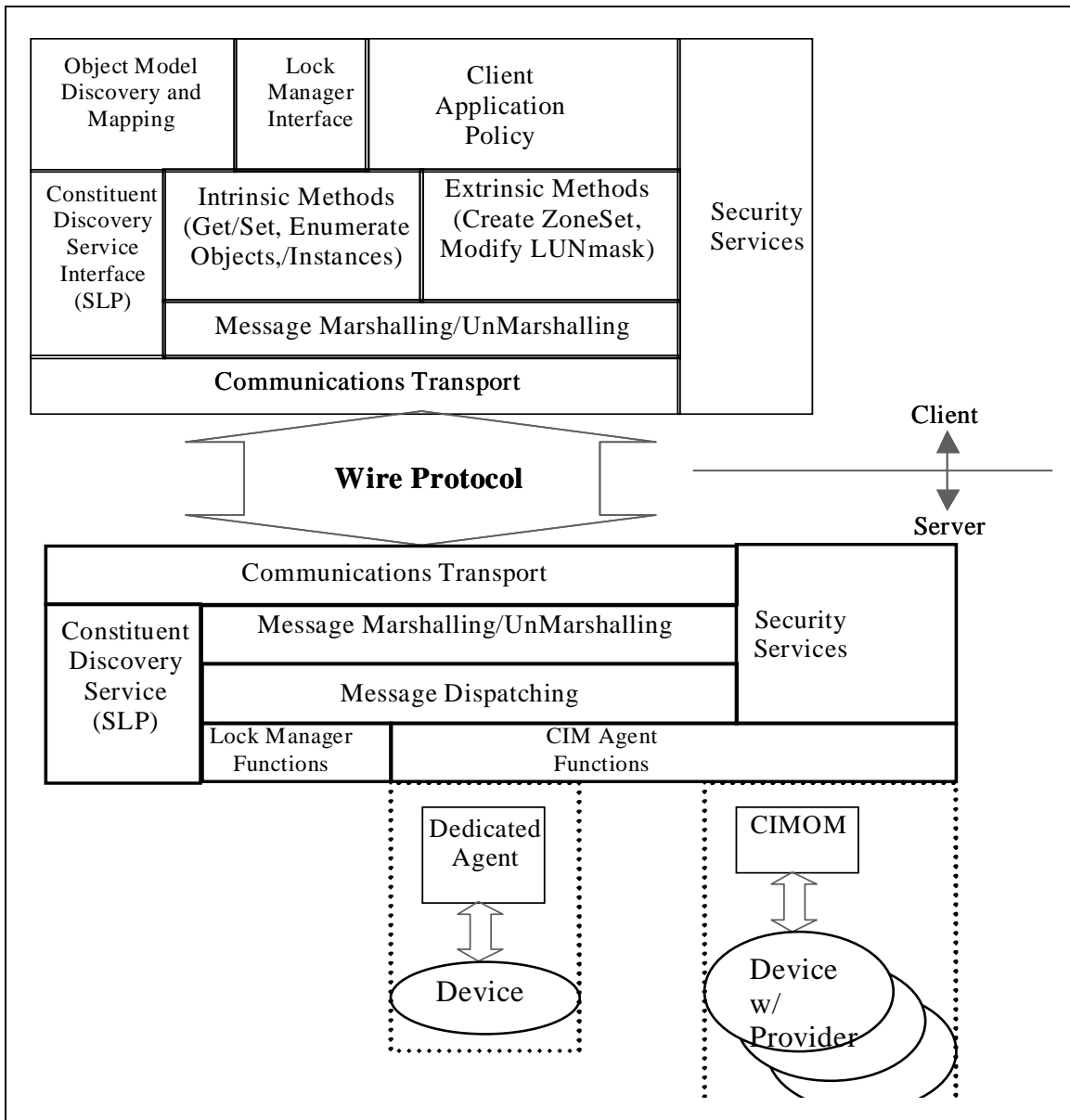
Object Model Discovery and Mapping	Lock Manager Interface	Client Application Policy	
Constituent Discovery Service Interface (SLP)	Intrinsic Methods (Get/Set, Enumerate Objects,/Instances)	Extrinsic Methods (Create ZoneSet, Modify LUNmask)	Security Services
	Message Marshalling/UnMarshalling		
Communications Transport			

Wire Protocol

Client

Server

Communications Transport	
Constituent Discovery Service (SLP)	Message Marshalling/UnMarshalling
	Message Dispatching
	Lock Manager Functions / CIM Agent Functions

Dedicated Agent

Device

CIMOM

Device w/ Provider

Figure 10 - Operational Environment

7.2   Using this Specification

This specification is insufficient as a single resource for the developers of SMI-S clients and servers. Developers are encouraged to first read the DMTF specifications on CIM and WBEM, as well as obtaining familiarity with UML and the IETF specification on Service Location Protocol (SLP).

A developer implementing SMI-S clients/servers should read this specification in sequence noting that *Storage Management Technical Specification, Part 3 Common Profiles, 1.6.1 Rev 6:* 4 Profile Introduction is intended principally as a reference relative to the particular device type that is being provided or managed in an SMI-S environment.

44

7.3   Language Bindings

As a messaging interface, this specification places no explicit requirements for syntax or grammar on the procedure call mechanisms employed to convert SMI-S messages into semantics consumable by modern programming languages. The syntax and grammar used to express these semantics is left at the discretion of each SMI-S developer.

Several open-source codebases are available for programmers who wish to streamline the task of parsing SMI-S messages into traditional procedure call semantics and using these semantics to store object instances. Consult the WBEMsource initiative (http://wbemsource.org) for current language bindings available to implement the SMI-S interface.