Document Number: DSP0825

Date: 2009-06-04

Version: 1.0.0

Shared Device Management Profile SM CLP Mapping Specification

Document Type: Specification

Document Status: DMTF Standard

Document Language: E

CONTENTS

Tables

Foreword

The *Shared Device Management Profile SM CLP Mapping Specification* (DSP0825) was prepared by the Server Management Working Group.

Conventions

The pseudo-code conventions utilized in this document are the Recipe Conventions as defined in SNIA SMI-S 1.1.0, section 7.6.

Acknowledgements

The authors wish to acknowledge the following participants from the DMTF Server Management Working Group:

- Aaron Merkin – IBM
- Jon Hass – Dell
- Khachatur Papanyan – Dell
- Enoch Suen – Dell
- Jeff Hilland – HP
- Christina Shaw – HP
- Perry Vincent – Intel
- John Leung – Intel

Introduction

This document defines the SM CLP mapping for CIM elements described in the *Shared Device Management Profile*. The information in this specification, combined with the *SM CLP-to-CIM Common Mapping Specification 1.0* (DSP0216), is intended to be sufficient to implement SM CLP commands relevant to the classes, properties and methods described in the *Shared Device Management Profile* using CIM operations.

The target audience for this specification is implementers of the SM CLP support for the *Shared Device Management Profile*.

Shared Device Management Profile SM CLP Mapping Specification

1 Scope

This specification contains the requirements for an implementation of the SM CLP to provide access to, and implement the behaviors of, the *Shared Device Management Profile*.

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

DMTF DSP1021, *Shared Device Management Profile 1.0*, http://www.dmtf.org/standards/published_documents/DSP1021_1.0.pdf

DMTF DSP0216, *SM CLP-to-CIM Common Mapping Specification 1.0*, http://www.dmtf.org/standards/published_documents/DSP0216_1.0.pdf

SNIA, *Storage Management Initiative Specification (SMI-S) 1.1.0*, http://www.snia.org/tech_activities/standards/curr_standards/smi

2.2 Other References

ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*, http://isotc.iso.org/livelink/livelink.exe?func=ll&objId=4230456&objAction=browse&sort=subtype

3 Terms and Definitions

For the purposes of this document, the following terms and definitions apply.

3.1
can
used for statements of possibility and capability, whether material, physical, or causal

3.2
cannot
used for statements of possibility and capability, whether material, physical or causal

3.3
conditional
indicates requirements to be followed strictly in order to conform to the document when the specified conditions are met

3.4

mandatory

indicates requirements to be followed strictly in order to conform to the document and from which no deviation is permitted

3.5

may

indicates a course of action permissible within the limits of the document

3.6

need not

indicates a course of action permissible within the limits of the document

3.7

optional

indicates a course of action permissible within the limits of the document

3.8

shall

indicates requirements to be followed strictly in order to conform to the document and from which no deviation is permitted

3.9

shall not

indicates requirements to be followed strictly in order to conform to the document and from which no deviation is permitted

3.10

should

indicates that among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others, or that a certain course of action is preferred but not necessarily required

3.11

should not

indicates that a certain possibility or course of action is deprecated but not prohibited

4 Symbols and Abbreviated Terms

The following symbols and abbreviations are used in this document.

4.1

CIM

Common Information Model

4.2

CLP

Command Line Protocol

4.3

DMTF

Distributed Management Task Force

4.4
IETF
Internet Engineering Task Force

4.5
SM
Server Management

4.6
SMI-S
Storage Management Initiative Specification

4.7
SNIA
Storage Networking Industry Association

5 Recipes

The following is a list of the common recipes used by the mappings in this specification. For a definition of each recipe, see DSP0216.

- smOpAssociators
- smOpReferences
- smOpInvokeMethod
- smShowAssociationInstances
- smResetRSC
- smRequestStateChange
- smShowInstance
- &smEnd
- smProcessError
- smShowInstances
- smShowInstanceWithReferenceProperties
- smShowInstancesWithReferenceProperties

For convenience, Table 1 lists each recipe defined in this mapping which is used for more than one verb or class mapping.

Table 1 – Local Recipes

Recipe Name	Description	Definition
lAddReferencedProperties	Add associated property to an instance of CIM_LogicalDevice.	See 5.1.

The following sections detail Local Recipes defined for use in this mapping.

5.1 lAddReferencedProperties

5.1.1 Description

Add the relevant associated properties to the instance of CIM_LogicalDevice.

5.1.2 Preconditions

`$device` contains the instance of CIM_LogicalDevice to which associated properties should be added.

`#all` is true if the "-all" option was specified with the command; otherwise, `#all` is false.

5.1.3 Pseudo Code

```
sub lAddReferencedProperties($device, #ReferencedPropertyNames) {
   #propertylist[] = NULL;
#Error = &smOpAssociators($device.GetObjectPath(),
   "CIM_SharingDependency",
   NULL,
   NULL,
   { "CurrentAccess" },
   $associations);
if (0 != #Error.code) {
      &smProcessOpError (#Error);
      //includes &smEnd;
}
   //only one allowed to be associated
   $sharingdependency = $associations[0];
   $device.CurrentAccess = $sharingDependency.CurrentAccess;
   #ReferencedPropertyNames = { "CurrentAccess" };
} //lAddReferencedProperties()
```

6 Mappings

The following sections detail the mapping of CLP verbs to CIM Operations for each CIM class defined in the *Shared Device Management Profile*.

6.1 CIM_ElementCapabilities

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 2 lists each SM CLP verb, the required level of support for the verb in conjunction with the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 2 is for informational purposes only; in case of a conflict between Table 2 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 2.

Table 2 – Command Verb Requirements for CIM_ElementCapabilities

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.1.2.
start	Not supported	
stop	Not supported	

No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`, `reset`, `set`, `start`, and `stop`.

6.1.1 Ordering of Results

When results are returned for multiple instances of CIM_ElementCapabilities, implementations shall utilize the following algorithm to produce the natural (that is, default) ordering:

- Results for CIM_ElementCapabilities are unordered; therefore, no algorithm is defined.

6.1.2 Show

This section describes how to implement the `show` verb when applied to an instance of CIM_ElementCapabilities. Implementations shall support the use of the `show` verb with CIM_ElementCapabilities.

The `show` command is used to display information about the CIM_ElementCapabilities instance or instances.

6.1.2.1 Show Multiple Instances – CIM_DeviceSharingCapabilities Reference

This command form is for the `show` verb applied to multiple instances. This command form corresponds to a `show` command issued against CIM_ElementCapabilities where the reference specified is to an instance of CIM_DeviceSharingCapabilities.

6.1.2.1.1 Command Form

```
show <CIM_ElementCapabilities single instance>
```

6.1.2.1.2 CIM Requirements

See CIM_ElementCapabilities in the "CIM Elements" section of the *Shared Device Management Profile* for the list of mandatory properties.

6.1.2.1.3 Behavior Requirements

6.1.2.1.3.1 Preconditions

`$instance` contains the instance of CIM_DeviceSharingCapabilities which is referenced by CIM_ElementCapabilities.

6.1.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ElementCapabilities”, $instance.getObjectPath() );
&smEnd;
```

6.1.2.2 Show Multiple Instances – CIM_EnabledLogicalElementCapabilities Reference

This command form is for the `show` verb applied to multiple instances. This command form corresponds to a `show` command issued against CIM_ElementCapabilities where the reference specified is to an instance of CIM_EnabledLogicalElementCapabilities.

6.1.2.2.1 Command Form

```
show <CIM_ElementCapabilities single instance>
```

6.1.2.2.2 CIM Requirements

See CIM_ElementCapabilities in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.1.2.2.3 Behavior Requirements

6.1.2.2.3.1 Preconditions

`$instance` contains the instance of CIM_EnabledLogicalElementCapabilities which is referenced by CIM_ElementCapabilities.

6.1.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ElementCapabilities”, $instance.getObjectPath() );
&smEnd;
```

6.1.2.3 Show a Single Instance – CIM_LogicalDevice Reference

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_ElementCapabilities where the reference specified is to an instance of CIM_LogicalDevice. An instance of CIM_LogicalDevice is referenced by exactly one instance of CIM_ElementCapabilities. Therefore, a single instance will be returned.

6.1.2.3.1 Command Form

```
show <CIM_ElementCapabilities single instance>
```

6.1.2.3.2 CIM Requirements

See CIM_ElementCapabilities in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.1.2.3.3 Behavior Requirements

6.1.2.3.3.1 Preconditions

`$instance` contains the instance of CIM_LogicalDevice which is referenced by CIM_ElementCapabilities.

6.1.2.3.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ElementCapabilities”, $instance.getObjectPath() );
&smEnd;
```

6.1.2.4 Show a Single Instance – CIM_SharedDeviceManagementService Reference

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_ElementCapabilities where the reference specified is to an instance of CIM_SharedDeviceManagementService. An instance of CIM_SharedDeviceManagementService is referenced by exactly one instance of CIM_ElementCapabilities. Therefore, a single instance will be returned.

6.1.2.4.1 Command Form

```
show <CIM_ElementCapabilities single instance>
```

6.1.2.4.2 CIM Requirements

See CIM_ElementCapabilities in the "CIM Elements" section of the *Shared Device Management Profile* for the list of mandatory properties.

6.1.2.4.3 Behavior Requirements

6.1.2.4.3.1 Preconditions

`$instance` contains the instance of CIM_SharedDeviceManagementService which is referenced by CIM_ElementCapabilities.

6.1.2.4.3.2 Pseudo Code

```
&smShowAssociationInstances ( "CIM_ElementCapabilities", $instance.getObjectPath() );
&smEnd;
```

6.1.2.5 Show a Single Instance – Both References (CIM_DeviceSharingCapabilities)

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_ElementCapabilities where both references are specified and therefore the desired instance is unambiguously identified.

6.1.2.5.1 Command Form

```
show <CIM_ElementCapabilities single instance>
```

6.1.2.5.2 CIM Requirements

See CIM_ElementCapabilities in the "CIM Elements" section of the *Shared Device Management Profile* for the list of mandatory properties.

6.1.2.5.3 Behavior Requirements

6.1.2.5.3.1 Preconditions

`$instanceA` contains the instance of CIM_DeviceSharingCapabilities which is referenced by CIM_ElementCapabilities.

`$instanceB` contains the instance of CIM_LogicalDevice which is referenced by CIM_ElementCapabilities.

6.1.2.5.3.2 Pseudo Code

```
&smShowAssociationInstance ( "CIM_ElementCapabilities", $instanceA.getObjectPath(),
   $instanceB.getObjectPath() );
&smEnd;
```

6.1.2.6 Show a Single Instance – Both References (CIM_EnabledLogicalElementCapabilities)

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_ElementCapabilities where both references are specified and therefore the desired instance is unambiguously identified.

6.1.2.6.1 Command Form

```
show <CIM_ElementCapabilities single instance>
```

6.1.2.6.2 CIM Requirements

See CIM_ElementCapabilities in the "CIM Elements" section of the *Shared Device Management Profile* for the list of mandatory properties.

6.1.2.6.3 Behavior Requirements

6.1.2.6.3.1 Preconditions

`$instanceA` contains the instance of CIM_EnabledLogicalElementCapabilities which is referenced by CIM_ElementCapabilities.

`$instanceB` contains the instance of CIM_SharedDeviceManagementService which is referenced by CIM_ElementCapabilities.

6.1.2.6.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_ElementCapabilities”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath() );
&smEnd;
```

6.2 CIM_EnabledLogicalElementCapabilities

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 3 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 3 is for informational purposes only; in case of a conflict between Table 3 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 3.

Table 3 – Command Verb Requirements for CIM_EnabledLogicalElementCapabilities

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.2.2.
show	Shall	See 6.2.3.
start	Not supported	
stop	Not supported	

No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`, `reset`, `set`, `start`, and `stop`.

6.2.1 Ordering of Results

When results are returned for multiple instances of CIM_EnabledLogicalElementCapabilities, implementations shall utilize the following algorithm to produce the natural (that is, default) ordering:

- Results for CIM_EnabledLogicalElementCapabilities are unordered; therefore, no algorithm is defined.

6.2.2 Set

This section describes how to implement the `set` verb when applied to an instance of CIM_EnabledLogicalElementCapabilities. Implementations may support the use of the `set` verb with CIM_EnabledLogicalElementCapabilities.

6.2.2.1 General Usage of Set for a Single Property

This command form corresponds to the general usage of the `set` verb to modify a single property of a target instance. This is the most common case.

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Shared Device Management Profile*.

6.2.2.1.1 Command Form

```
set <CIM_EnabledLogicalElementCapabilities single instance>
   <propertyname>=<propertyvalue>
```

6.2.2.1.2 CIM Requirements

See CIM_EnabledLogicalElementCapabilities in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.2.2.1.3 Behavior Requirements

```
$instance=<CIM_EnabledLogicalElementCapabilities single instance>
#propertyNames[] = {<propertyname>};
#propertyValues[] = {<propertyvalue>};
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.2.2.2 General Usage of Set for Multiple Properties

This command form corresponds to the general usage of the `set` verb to modify multiple properties of a target instance where there is not an explicit relationship between the properties. This is the most common case.

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Shared Device Management Profile*.

6.2.2.2.1 Command Form

```
set <CIM_EnabledLogicalElementCapabilities single instance>
   <propertyname1>=<propertyvalue1><propertynamen>=<propertyvaluen>
```

6.2.2.2.2 CIM Requirements

See CIM_EnabledLogicalElementCapabilities in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.2.2.2.3 Behavior Requirements

```
$instance=<CIM_EnabledLogicalElementCapabilities single instance>
#propertyNames[] = {<propertyname>};
for #i < n
   {
   #propertyNames[#i] = <propertname#i>
   #propertyValues[#i] = <propertyvalue#i>
   }
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.2.3 Show

This section describes how to implement the show verb when applied to an instance of CIM_EnabledLogicalElementCapabilities. Implementations shall support the use of the show verb with CIM_EnabledLogicalElementCapabilities.

The show verb is used to display information about an instance or instances of the CIM_EnabledLogicalElementCapabilities class.

6.2.3.1 Show a Single Instance

This command form is for the show verb applied to a single instance of CIM_EnabledLogicalElementCapabilities.

6.2.3.1.1 Command Form

```
show <CIM_EnabledLogicalElementCapabilities single instance>
```

6.2.3.1.2 CIM Requirements

See CIM_EnabledLogicalElementCapabilities in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.2.3.1.3 Behavior Requirements

6.2.3.1.3.1 Preconditions

#all is true if the “-all” option was specified with the command; otherwise, #all is false.

6.2.3.1.3.2 Pseudo Code

```
$instance=<CIM_EnabledLogicalElementCapabilities single instance>
#propertylist[] = NULL;
if ( false == #all )
   {
   #propertylist[] = {//all mandatory non-key properties}
   }
&smShowInstance ( $instance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.2.3.2 Show Multiple Instances

This command form is for the `show` verb applied to multiple instances of CIM_EnabledLogicalElementCapabilities. This command form corresponds to UFsT-based selection within a capabilities collection.

6.2.3.2.1 Command Form

```
show <CIM_EnabledLogicalElementCapabilities multiple objects>
```

6.2.3.2.2 CIM Requirements

See CIM_EnabledLogicalElementCapabilities in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.2.3.2.3 Behavior Requirements

6.2.3.2.3.1 Preconditions

`$containerInstance` contains the instance of CIM_ConcreteCollection for which the contained CIM_Capabilities instances are displayed. CIM_Capabilities instances are addressed via an aggregating instance of CIM_ConcreteCollection.

`#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

6.2.3.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
   #propertylist[] = {//all mandatory non-key properties};
   }
&smShowInstances ( “CIM_EnabledLogicalElementCapabilities”, “CIM_MemberOfCollection”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.3 CIM_HostedService

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 4 lists each SM CLP verb, the required level of support for the verb in conjunction with the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 4 is for informational purposes only; in case of a conflict between Table 4 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 4.

Table 4 – Command Verb Requirements for CIM_HostedService

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	

Command Verb	Requirement	Comments
set	Not supported	
show	Shall	See 6.3.2.
start	Not supported	
stop	Not supported	

No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`, `reset`, `set`, `start`, and `stop`.

6.3.1 Ordering of Results

When results are returned for multiple instances of CIM_HostedService, implementations shall utilize the following algorithm to produce the natural (that is, default) ordering:

- Results for CIM_HostedService are unordered; therefore, no algorithm is defined.

6.3.2 Show

This section describes how to implement the `show` verb when applied to an instance of CIM_HostedService. Implementations shall support the use of the `show` verb with CIM_HostedService.

The `show` command is used to display information about the CIM_HostedService instance or instances.

6.3.2.1 Show Multiple Instances – CIM_ComputerSystem Reference

This command form is for the `show` verb applied to multiple instances. This command form corresponds to a `show` command issued against CIM_HostedService where only one reference is specified and the reference is to an instance of CIM_ComputerSystem.

6.3.2.1.1 Command Form

```
show <CIM_HostedService multiple objects>
```

6.3.2.1.2 CIM Requirements

See CIM_HostedService in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.3.2.1.3 Behavior Requirements

6.3.2.1.3.1 Preconditions

`$instance` contains the instance of CIM_ComputerSystem which is referenced by CIM_HostedService.

6.3.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_HostedService”, $instance.getObjectPath() );
&smEnd;
```

6.3.2.2 Show a Single Instance – CIM_SharedDeviceManagementService Reference

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_HostedService where the reference specified is to an instance of CIM_SharedDeviceManagementService. An instance of CIM_SharedDeviceManagementService is referenced by exactly one instance of CIM_HostedService. Therefore, a single instance will be returned.

6.3.2.2.1 Command Form

```
show <CIM_HostedService single instance>
```

6.3.2.2.2 CIM Requirements

See CIM_HostedService in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.3.2.2.3 Behavior Requirements

6.3.2.2.3.1 Preconditions

`$instance` contains the instance of CIM_SharedDeviceManagementService which is referenced by CIM_HostedService.

6.3.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_HostedService”, $instance.getObjectPath() );
&smEnd;
```

6.3.2.3 Show a Single Instance – Both References

This command form is for the `show` verb applied to a single instance. This command form corresponds to a show command issued against CIM_HostedService where both references are specified and therefore the desired instance is unambiguously identified.

6.3.2.3.1 Command Form

```
show <CIM_HostedService single instance>
```

6.3.2.3.2 CIM Requirements

See CIM_HostedService in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.3.2.3.3 Behavior Requirements

6.3.2.3.3.1 Preconditions

`$instanceA` contains the instance of CIM_ComputerSystem which is referenced by CIM_HostedService.

`$instanceB` contains the instance of CIM_SharedDeviceManagementService which is referenced by CIM_HostedService.

6.3.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_HostedService”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath() );
&smEnd;
```

6.4 CIM_ServiceAffectsElement

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 5 lists each SM CLP verb, the required level of support for the verb in conjunction with the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 5 is for informational purposes only; in case of a conflict between Table 5 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 5.

Table 5 – Command Verb Requirements for CIM_ServiceAffectsElement

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.4.2.
start	Not supported	
stop	Not supported	

No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`, `reset`, `set`, `start`, and `stop`.

6.4.1 Ordering of Results

When results are returned for multiple instances of CIM_ServiceAffectsElement, implementations shall utilize the following algorithm to produce the natural (that is, default) ordering:

- Results for CIM_ServiceAffectsElement are unordered; therefore, no algorithm is defined.

6.4.2 Show

This section describes how to implement the `show` verb when applied to an instance of CIM_ServiceAffectsElement. Implementations shall support the use of the `show` verb with CIM_ServiceAffectsElement.

The `show` command is used to display information about the CIM_ServiceAffectsElement instance or instances.

6.4.2.1 Show Multiple Instances – CIM_SharedDeviceManagementService Reference

This command form is for the `show` verb applied to multiple instances. This command form corresponds to a `show` command issued against CIM_ServiceAffectsElement where only one reference is specified and the reference is to an instance of CIM_SharedDeviceManagementService.

6.4.2.1.1 Command Form

```
show <CIM_ServiceAffectsElement multiple objects>
```

6.4.2.1.2 CIM Requirements

See CIM_ServiceAffectsElement in the "CIM Elements" section of the *Shared Device Management Profile* for the list of mandatory properties.

6.4.2.1.3 Behavior Requirements

6.4.2.1.3.1 Preconditions

`$instance` contains the instance of CIM_SharedDeviceManagementService which is referenced by CIM_ServiceAffectsElement.

`#all` is true if the "-all" option was specified with the command; otherwise, `#all` is false.

6.4.2.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
      #propertylist[] = {//all mandatory non-key properties};
   }
&smShowAssociationInstances ( "CIM_ServiceAffectsElement", $instance.getObjectPath(),
   #propertylist[] );
&smEnd;
```

6.4.2.2 Show Multiple Instances – CIM_LogicalDevice Reference

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_ServiceAffectsElement where the reference specified is to an instance of CIM_LogicalDevice.

6.4.2.2.1 Command Form

```
show <CIM_ServiceAffectsElement single instance>
```

6.4.2.2.2 CIM Requirements

See CIM_ServiceAffectsElement in the "CIM Elements" section of the *Shared Device Management Profile* for the list of mandatory properties. Section

6.4.2.2.3 Behavior Requirements

6.4.2.2.3.1 Preconditions

`$instance` contains the instance of CIM_LogicalDevice which is referenced by CIM_ServiceAffectsElement.

`#all` is true if the "-all" option was specified with the command; otherwise, `#all` is false.

6.4.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
      #propertylist[] = {//all mandatory non-key properties};
   }
&smShowAssociationInstances ( "CIM_ServiceAffectsElement", $instance.getObjectPath(),
   #propertylist[] );
&smEnd;
```

6.4.2.3 Show a Single Instance – Both References

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_ServiceAffectsElement where both references are specified and therefore the desired instance is unambiguously identified.

6.4.2.3.1 Command Form

```
show <CIM_ServiceAffectsElement single instance>
```

6.4.2.3.2 CIM Requirements

See CIM_ServiceAffectsElement in the "CIM Elements" section of the *Shared Device Management Profile* for the list of mandatory properties.

6.4.2.3.3 Behavior Requirements

6.4.2.3.3.1 Preconditions

`$instanceA` contains the instance of CIM_ServiceAvailableToElement which is referenced by CIM_ServiceAffectsElement.

`$instanceB` contains the instance of CIM_LogicalDevice which is referenced by CIM_ServiceAffectsElement.

`#all` is true if the "-all" option was specified with the command; otherwise, `#all` is false.

6.4.2.3.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
      #propertylist[] = {//all mandatory non-key properties};
   }
&smShowAssociationInstance ( "CIM_ServiceAffectsElement", $instanceA.getObjectPath(),
   $instanceB.getObjectPath(), #propertylist[] );
&smEnd;
```

6.5 CIM_SharedDeviceManagementService

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 6 lists each SM CLP verb, the required level of support for the verb in conjunction with the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 6 is for informational purposes only; in case of a conflict between Table 6 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 6.

Table 6 – Command Verb Requirements for CIM_SharedDeviceManagementService

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	

Command Verb	Requirement	Comments
reset	May	See 6.5.2.
set	May	See 6.5.3.
show	Shall	See 6.5.4.
start	May	See 6.5.5.
stop	May	See 6.5.6.

No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, and `load`.

6.5.1 Ordering of Results

When results are returned for multiple instances of CIM_SharedDeviceManagementService, implementations shall utilize the following algorithm to produce the natural (that is, default) ordering:

- Results for CIM_SharedDeviceManagementService are unordered; therefore, no algorithm is defined.

6.5.2 Reset

This section describes how to implement the `reset` verb when applied to an instance of CIM_SharedDeviceManagementService. Implementations may support the use of the `reset` verb with CIM_SharedDeviceManagementService.

The `reset` verb is used to initiate a reset of the CIM_SharedDeviceManagementService.

6.5.2.1 Reset a Single Instance

This command form is for the initiation of a reset action against a single instance of CIM_SharedDeviceManagementService. The mapping is implemented as an invocation of the RequestStateChange() method on the instance.

6.5.2.1.1 Command Form

```
reset <CIM_SharedDeviceManagementService single instance>
```

6.5.2.1.2 CIM Requirements

```
uint16 EnabledState;
uint16 RequestedState;
uint32 EnabledLogicalElement.RequestStateChange (
   [IN] uint16 RequestedState,
   [OUT] REF CIM_ConcreteJob Job,
   [IN] datetime TimeoutPeriod );
```

6.5.2.1.3 Behavior Requirements

```
$instance=<CIM_SharedDeviceManagementService single instance>
&smResetRSC ( $instance.getObjectPath() );
&smEnd;
```

6.5.3 Set

This section describes how to implement the `set` verb when applied to an instance of CIM_SharedDeviceManagementService. Implementations may support the use of the `set` verb with CIM_SharedDeviceManagementService.

6.5.3.1 General Usage of Set for a Single Property

This command form corresponds to the general usage of the set verb to modify a single property of a target instance. This is the most common case.

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Shared Device Management Profile*.

6.5.3.1.1 Command Form

```
set <CIM_SharedDeviceManagementService single instance> <propertyname>=<propertyvalue>
```

6.5.3.1.2 CIM Requirements

See CIM_SharedDeviceManagementService in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.5.3.1.3 Behavior Requirements

```
$instance=<CIM_SharedDeviceManagementService single instance>
#propertyNames[] = {<propertyname>};
#propertyValues[] = {<propertyvalue>};
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.5.3.2 General Usage of Set for Multiple Properties

This command form corresponds to the general usage of the `set` verb to modify multiple properties of a target instance where there is not an explicit relationship between the properties. This is the most common case.

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Shared Device Management Profile*.

6.5.3.2.1 Command Form

```
set <CIM_SharedDeviceManagementService single instance>
   <propertyname1>=<propertyvalue1> <propertynamen>=<propertyvaluen>
```

6.5.3.2.2 CIM Requirements

See CIM_SharedDeviceManagementService in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.5.3.2.3 Behavior Requirements

```
$instance=<CIM_SharedDeviceManagementService single instance>
#propertyNames[] = {<propertyname>};
for #i < n
   {
   #propertyNames[#i] = <propertname#i>
   #propertyValues[#i] = <propertyvalue#i>
   }
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.5.4 Show

This section describes how to implement the `show` verb when applied to an instance of CIM_SharedDeviceManagementService. Implementations shall support the use of the `show` verb with CIM_SharedDeviceManagementService.

The `show` verb is used to display information about the CIM_SharedDeviceManagementService instance.

6.5.4.1 Show a Single Instance

This command form is for the `show` verb applied to a single instance of CIM_SharedDeviceManagementService.

6.5.4.1.1 Command Form

```
show <CIM_SharedDeviceManagementService single instance>
```

6.5.4.1.2 CIM Requirements

See CIM_SharedDeviceManagementService in the "CIM Elements" section of the *Shared Device Management Profile* for the list of mandatory properties.

6.5.4.1.3 Behavior Requirements

6.5.4.1.3.1 Preconditions

`#all` is true if the "-all" option was specified with the command; otherwise, `#all` is false.

6.5.4.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
      #propertylist[] = {//all mandatory non-key properties};
   }
$instance=<CIM_SharedDeviceManagementService single instance>
&smShowInstance ( $instance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.5.4.2 Show Multiple Instances

This command form is for the `show` verb applied to multiple instances of CIM_SharedDeviceManagementService. This command form corresponds to UFsT-based selection within a scoping system.

6.5.4.2.1 Command Form

```
show <CIM_SharedDeviceManagementService multiple objects>
```

6.5.4.2.2 CIM Requirements

See CIM_SharedDeviceManagementService in the "CIM Elements" section of the *Shared Device Management Profile* for the list of mandatory properties.

6.5.4.2.3 Behavior Requirements

6.5.4.2.3.1 Preconditions

`$containerInstance` contains the instance of CIM_ComputerSystem for which the scoped CIM_SharedDeviceManagementService instances are displayed. The *Shared Device Management Profile* requires that the CIM_SharedDeviceManagementService instance be associated with its scoping system via an instance of the CIM_HostedService association.

`#all` is true if the "-all" option was specified with the command; otherwise, `#all` is false.

6.5.4.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
   #propertylist[] = {//all mandatory non-key properties};
   }
&smShowInstances ( "CIM_SharedDeviceManagementService", "CIM_HostedService",
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.5.5 Start

This section describes how to implement the `start` verb when applied to an instance of CIM_SharedDeviceManagementService. Implementations may support the use of the `start` verb with CIM_SharedDeviceManagementService.

The `start` verb is used to enable the device sharing functionality represented by the CIM_SharedDeviceManagementService.

6.5.5.1 Start a Single Instance

This command form is for the `start` verb applied to a single instance of CIM_SharedDeviceManagementService.

6.5.5.1.1 Command Form

```
start <CIM_SharedDeviceManagementService single instance>
```

6.5.5.1.2 CIM Requirements

```
uint16 EnabledState;
uint16 RequestedState;
uint32 EnabledLogicalElement.RequestStateChange (
   [IN] uint16 RequestedState,
   [OUT] REF CIM_ConcreteJob Job,
   [IN] datetime TimeoutPeriod );
```

6.5.5.1.3 Behavior Requirements

```
$instance=<CIM_SharedDeviceManagementService single instance>
&smStartRSC ( $instance.getObjectPath() );
&smEnd;
```

6.5.6 Stop

This section describes how to implement the `stop` verb when applied to an instance of CIM_SharedDeviceManagementService. Implementations may support the use of the `stop` verb with CIM_SharedDeviceManagementService.

The `stop` verb is used to disable the functionality represented by the CIM_SharedDeviceManagementService instance.

6.5.6.1 Stop a Single Instance

This command form is for the `stop` verb applied to a single instance of CIM_SharedDeviceManagementService.

6.5.6.1.1 Command Form

```
stop <CIM_SharedDeviceManagementService single instance>
```

6.5.6.1.2 CIM Requirements

```
uint16 EnabledState;
uint16 RequestedState;
uint32 EnabledLogicalElement.RequestStateChange (
   [IN] uint16 RequestedState,
   [OUT] REF CIM_ConcreteJob Job,
   [IN] datetime TimeoutPeriod );
```

6.5.6.1.3 Behavior Requirements

```
$instance=<CIM_SharedDeviceManagementService single instance>
&smStopRSC ( $instance.getObjectPath() );
&smEnd;
```

6.6 CIM_SharingDependency

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 7 lists each SM CLP verb, the required level of support for the verb in conjunction with the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 7 is for informational purposes only; in case of a conflict between Table 7 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 7.

Table 7 – Command Verb Requirements for CIM_SharingDependency

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	

Command Verb	Requirement	Comments
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.6.2.
start	Not supported	
stop	Not supported	

No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`, `reset`, `start`, and `stop`.

6.6.1 Ordering of Results

When results are returned for multiple instances of CIM_SharingDependency, implementations shall utilize the following algorithm to produce the natural (that is, default) ordering:

- Results for CIM_SharingDependency are unordered; therefore, no algorithm is defined.

6.6.2 Show

This section describes how to implement the `show` verb when applied to an instance of CIM_SharingDependency. Implementations shall support the use of the `show` verb with CIM_SharingDependency.

The `show` command is used to display information about the CIM_SharingDependency instance or instances.

6.6.2.1 Show Multiple Instances – Real Logical Device

This command form is for the `show` verb applied to multiple instances. This command form corresponds to a `show` command issued against CIM_SharingDependency where only one reference is specified and the reference is to an instance of CIM_LogicalDevice which is a Real Logical Device.

6.6.2.1.1 Command Form

```
show <CIM_SharingDependency multiple objects>
```

6.6.2.1.2 CIM Requirements

See CIM_SharingDependency in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.6.2.1.3 Behavior Requirements

6.6.2.1.3.1 Preconditions

`$instance` contains the instance of CIM_ComputerSystem which is referenced by CIM_SharingDependency.

`#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

6.6.2.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
      #propertylist[] = {//all mandatory non-key properties};
   }
&smShowAssociationInstances ( “CIM_SharingDependency”, $instance.getObjectPath(),
   #propertylist[] );
&smEnd;
```

6.6.2.2 Show a Single Instance – Sharing Logical Device Reference

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_SharingDependency where the reference specified is to an instance of CIM_LogicalDevice which is a Sharing Logical Device. A Sharing Logical Device is referenced by exactly one instance of CIM_SharingDependency. Therefore, a single instance will be returned.

6.6.2.2.1 Command Form

```
show <CIM_SharingDependency single instance>
```

6.6.2.2.2 CIM Requirements

See CIM_SharingDependency in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.6.2.2.3 Behavior Requirements

6.6.2.2.3.1 Preconditions

`$instance` contains the instance of CIM_LogicalDevice which is referenced by CIM_SharingDependency.

`#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

6.6.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
      #propertylist[] = {//all mandatory non-key properties};
   }
&smShowAssociationInstances ( “CIM_SharingDependency”, $instance.getObjectPath(),
   #propertylist[] );
&smEnd;
```

6.6.2.3 Show a Single Instance – Both References

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_SharingDependency where both references are specified and therefore the desired instance is unambiguously identified.

6.6.2.3.1 Command Form

```
show <CIM_SharingDependency single instance>
```

6.6.2.3.2 CIM Requirements

See CIM_SharingDependency in the "CIM Elements" section of the *Shared Device Management Profile* for the list of mandatory properties.

6.6.2.3.3 Behavior Requirements

6.6.2.3.3.1 Preconditions

`$instanceA` contains one of the instances of CIM_LogicalDevice which is referenced by CIM_SharingDependency.

`$instanceB` contains one of the instances of CIM_LogicalDevice which is referenced by CIM_SharingDependency.

`#all` is true if the "-all" option was specified with the command; otherwise, `#all` is false.

6.6.2.3.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
      #propertylist[] = {//all mandatory non-key properties};
   }
&smShowAssociationInstance ( “CIM_SharingDependency”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath(), #propertylist[] );
&smEnd;
```

6.7 CIM_DeviceSharingCapabilities

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 8 lists each SM CLP verb, the required level of support for the verb in conjunction with the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 8 is for informational purposes only; in case of a conflict between Table 8 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 8.

Table 8 – Command Verb Requirements for CIM_DeviceSharingCapabilities

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.7.2.
show	Shall	See 6.7.3.
start	Not supported	
stop	Not supported	

No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`, `reset`, `set`, `start`, and `stop`.

6.7.1 Ordering of Results

When results are returned for multiple instances of CIM_DeviceSharingCapabilities, implementations shall utilize the following algorithm to produce the natural (that is, default) ordering:

- Results for CIM_DeviceSharingCapabilities are unordered; therefore, no algorithm is defined.

6.7.2 Set

This section describes how to implement the `set` verb when applied to an instance of CIM_DeviceSharingCapabilities. Implementations may support the use of the `set` verb with CIM_DeviceSharingCapabilities.

6.7.2.1 General Usage of Set for a Single Property

This command form corresponds to the general usage of the `set` verb to modify a single property of a target instance. This is the most common case.

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Shared Device Management Profile*.

6.7.2.1.1 Command Form

```
set <CIM_DeviceSharingCapabilities single instance> <propertyname>=<propertyvalue>
```

6.7.2.1.2 CIM Requirements

See CIM_DeviceSharingCapabilities in the “CIM Elements” section of the *Shared Device Management Profile* for the list of modifiable properties.

6.7.2.1.3 Behavior Requirements

```
$instance=<CIM_DeviceSharingCapabilities single instance>
#propertyNames[] = {<propertyname>};
#propertyValues[] = {<propertyvalue>};
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.7.2.2 General Usage of Set for Multiple Properties

This command form corresponds to the general usage of the `set` verb to modify multiple properties of a target instance where there is not an explicit relationship between the properties. This is the most common case.

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Shared Device Management Profile*.

6.7.2.2.1 Command Form

```
set <CIM_DeviceSharingCapabilities single instance> <propertyname1>=<propertyvalue1>
   <propertynamen>=<propertyvaluen>
```

6.7.2.2.2 CIM Requirements

See CIM_DeviceSharingCapabilities in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.7.2.2.3 Behavior Requirements

```
$instance=<CIM_DeviceSharingCapabilities single instance>
#propertyNames[] = {<propertyname>};
for #i < n
   {
   #propertyNames[#i] = <propertname#i>
   #propertyValues[#i] = <propertyvalue#i>
   }
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.7.3 Show

This section describes how to implement the `show` verb when applied to an instance of CIM_DeviceSharingCapabilities. Implementations shall support the use of the `show` verb with CIM_DeviceSharingCapabilities.

The `show` command is used to display information about the CIM_DeviceSharingCapabilities instance or instances.

6.7.3.1 Show Multiple Instances – CIM_ConcreteCollection Reference

This command form is for the `show` verb applied to multiple instances. This command form corresponds to a `show` command issued against CIM_DeviceSharingCapabilities where only one reference is specified and the reference is to an instance of CIM_ConcreteCollection.

6.7.3.1.1 Command Form

```
show <CIM_DeviceSharingCapabilities multiple objects>
```

6.7.3.1.2 CIM Requirements

See CIM_DeviceSharingCapabilities in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.7.3.1.3 Behavior Requirements

6.7.3.1.3.1 Preconditions

`$containerInstance` contains the instance of CIM_ConcreteCollection for which we are displaying scoped CIM_DeviceSharingCapabilities instances.

`#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

6.7.3.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
   #propertylist[] = {//all mandatory non-key properties};
   }
&smShowInstances ( “CIM_DeviceSharingCapabilities”, “CIM_ElementCapabilities”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.7.3.2 Show a Single Instance

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against a single instance of CIM_DeviceSharingCapabilities.

6.7.3.2.1 Command Form

```
show <CIM_DeviceSharingCapabilities single instance>
```

6.7.3.2.2 CIM Requirements

See CIM_DeviceSharingCapabilities in the "CIM Elements" section of the *Shared Device Management Profile* for the list of mandatory properties.

6.7.3.2.3 Behavior Requirements

`#all` is true if the "-all" option was specified with the command; otherwise, `#all` is false.

6.7.3.2.3.1 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
      #propertylist[] = {//all mandatory non-key properties};
   }
$instance=<CIM_DeviceSharingCapabilities single instance>
&smShowInstance ( $instance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.8 CIM_LogicalDevice (Sharing Logical Device)

This section specifies the command verb requirements for the instance of a subclass of CIM_LogicalDevice which is defined to be a Sharing Logical Device in accordance with the *Shared Device Management Profile*.

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 9 lists each SM CLP verb, the required level of support for the verb in conjunction with the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 9 is for informational purposes only; in case of a conflict between Table 9 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 9.

Table 9 – Command Verb Requirements for CIM_LogicalDevice (Sharing Logical Device)

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	

Command Verb	Requirement	Comments
set	Shall	See 6.8.2.
show	Shall	See 6.8.3.
start	Not supported	
stop	Not supported	

No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, and `load`.

6.8.1 Ordering of Results

When results are returned for multiple instances of CIM_SharedDeviceManagementService, implementations shall utilize the following algorithm to produce the natural (that is, default) ordering:

- Results for CIM_SharedDeviceManagementService are unordered; therefore, no algorithm is defined.

6.8.2 Set

This section describes how to implement the `set` verb when applied to an instance of CIM_LogicalDevice.

6.8.2.1 Using Set to Modify Access to a Shared Device

This command form corresponds to using the `set` verb to assign a value to the currentaccess pseudo-property of the CIM_LogicalDevice instance. This results in the invocation of the ShareDevice method on an instance of CIM_SharedDeviceManagementService.

This command form shall be supported.

6.8.2.1.1 Command Form

```
set <CIM_LogicalDevice single instance> currentaccess=<requestvalue>
```

6.8.2.1.2 CIM Requirements

```
uint16 SharingDependency.CurrentAccess;
uint16 SharingDependency.OtherCurrentAccess;
uint32 SharedDeviceManagementService.ShareDevice (
   [IN] uint16 RequestedAccess,
   [IN} REF CIM_LogicalDevice Device,
   [IN] REF CIM_System System,
   [IN] datetime TimeoutPeriod,
   [IN] Boolean Force,
   [OUT] REF CIM_ConcreteJob Job);
```

6.8.2.1.3 Behavior Requirements

```
$instance=<CIM_LogicalDevice single instance>

//step 1 Find the target System instance
#Error = smOpAssociators (
   $instance.GetObjectPath(),
   “CIM_SystemDevice”,
   “CIM_System”,
   NULL,
   NULL,
   $Systems[]);
//Only one system is allowed to be associated via SystemDevice
$System-> = $Systems[0].getObjectPath();

//step 2 Find the LogicalDevice instance which reflects the real Logical Device
#Error = smOpAssociators(
   $instance.GetObjectPath(),
   “CIM_SharingDependency”,
   “CIM_LogicalDevice”,
   NULL,
   NULL,
   $RealDevices[]);
//Only one system is allowed to be associated via SharingDependency
$RealDevice-> = $RealDevices[0].getObjectPath();

// Step 3, find the SharedDeviceManagementService associated to the Real
// Logical Device
#Error = smOpAssociators(
   $RealDevice->,
   “CIM_ServiceAffectsElement”,
   “CIM_SharedDeviceManagementService”,
   NULL,
   NULL,
   $Services[]);
//Only one instance of SharedDeviceManagementService is allowed to be associated with
//the Real LogicalDevice
$Service-> = $Services[0].getObjectPath();

//Step 4, If Force option was specified, the Force parameter is true, assume the
   option exists and its boolean value is equivalent to whether or not it was
   specified
#ForceParameter = #ForceOption;

//Step 5, Assume a hard-coded default time. Implementations are not required
//to use this specific value.
#TimeoutPeriod = 30;

//Step 6, build parameter lists for method invocation
%InArguments[] = {newArgument(“RequestedAccess”, <requestvalue>),
   newArgument (“Device”, #Device),
   newArgument (“System”, #System),
   newArgument(“TimeoutPeriod”, #TimeoutPeriod),
   newArgument (“Force”, #ForceParameter)};
%OutArguments[] = { };

//step 7, invoke method
#Error = smOpInvokeMethod ($Service->,
   “ShareDevice”,
   %InArguments[],
   %OutArguments[],
   #returnStatus);
```

```
//step 8, process return code to CLP Command Status
if (0 != #Error.code)  {
   //method invocation failed
   if ( (NULL != #Error.$error) && (NULL != #Error.$error[0]) )    {
      //if the method invocation contains an embedded error
      //use it for the Error for the overall job
      &smAddError($job, #Error.$error[0]);
      &smMakeCommandStatus($job);
      &smEnd;
   }
   else {
      //operation failed, but no detailed error instance, need to make one up
      //make an Error instance and associate with job for Operation
         $OperationError = smNewInstance(“CIM_Error”);
         //CIM_ERR_FAILED
         $OperationError.CIMStatusCode = 1;
         //Software Error
         $OperationError.ErrorType = 4;
         //Unknown
         $OperationError.PerceivedSeverity = 0;
         $OperationError.OwningEntity = DMTF:SMCLP;
         $OperationError.MessageID = 0x00000009;
         $OperationError.Message = “An internal software error has occurred.”;
         &smAddError($job, $OperationError);
         &smMakeCommandStatus($job);
         &smEnd;
   }
}//if CIM op failed
else if (0 == #returnStatus)  {
   //completed successfully
   &smCommandCompleted($job);
   &smEnd;
}
else   {
   //unspecified return code, generic failure
   $OperationError = smNewInstance(“CIM_Error”);
   //CIM_ERR_FAILED
   $OperationError.CIMStatusCode = 1;
   //Other
   $OperationError.ErrorType = 1;
   //Low
   $OperationError.PerceivedSeverity = 2;
   $OperationError.OwningEntity = DMTF:SMCLP;
   $OperationError.MessageID = 0x00000002;
   $OperationError.Message = “Failed. No further information is available.”;
   &smAddError($job, $OperationError);
   &smMakeCommandStatus($job);
   &smEnd;
}
```

6.8.2.2 General Usage of Set for a Single Property

This command form corresponds to the general usage of the `set` verb to modify a single property of a target instance. This is the most common case.

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Shared Device Management Profile*.

6.8.2.2.1 Command Form

```
set <CIM_LogicalDevice single instance> <propertyname>=<propertyvalue>
```

6.8.2.2.2 CIM Requirements

See CIM_LogicalDevice in the “CIM Elements” section of the *Shared Device Management Profile* for the list of modifiable properties.

6.8.2.2.3 Behavior Requirements

```
$instance=<CIM_LogicalDevice single instance>
#propertyNames[] = {<propertyname>};
#propertyValues[] = {<propertyvalue>};
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.8.2.3 General Usage of Set for Multiple Properties

This command form corresponds to the general usage of the `set` verb to modify multiple properties of a target instance where there is not an explicit relationship between the properties. This is the most common case.

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Shared Device Management Profile*.

6.8.2.3.1 Command Form

```
set <CIM_LogicalDevice single instance> <propertyname1>=<propertyvalue1>
       <propertynamen>=<propertyvaluen>
```

6.8.2.3.2 CIM Requirements

See CIM_LogicalDevice in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.8.2.3.3 Behavior Requirements

```
$instance=<CIM_LogicalDevice single instance>
#propertyNames[] = {<propertyname>};
for #i < n
   {
   #propertyNames[#i] = <propertname#i>
   #propertyValues[#i] = <propertyvalue#i>
   }
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.8.3 Show

This section describes how to implement the `show` verb when applied to an instance of CIM_LogicalDevice which is a Sharing Logical Device. Implementations shall support the use of the `show` verb with CIM_LogicalDevice.

The `show` verb is used to display information about the CIM_LogicalDevice instance.

6.8.3.1 Show a Single Instance

This command form is for the `show` verb applied to a single instance of CIM_LogicalDevice.

6.8.3.1.1 Command Form

```
show <CIM_LogicalDevice single instance>
```

6.8.3.1.2 CIM Requirements

See CIM_LogicalDevice in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.8.3.1.3 Behavior Requirements

For the *Shared Device Management Profile*, a pseudo property is added to the Sharing Logical Device. The pseudo property has the same name and value as the CurrentAccess property on the referencing CIM_SharingDependency instance.

`#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

6.8.3.1.3.1 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
   #propertylist[] = { “ElementName” }
   }
$device=<CIM_LogicalDevice single instance>
&lAddReferencedProperties ( $device, $referencedPropertyNames[], #all );
&smShowInstanceWithReferencedProperties ( $device, #propertyList[],
   $referencedPropertyNames[] );
&smEnd;
```

6.8.3.2 Show Multiple Instances

This command form is for the `show` verb applied to multiple instances of CIM_LogicalDevice. This command form corresponds to UFsT-based selection within a scoping system.

6.8.3.2.1 Command Form

```
show <CIM_LogicalDevice multiple objects>
```

6.8.3.2.2 CIM Requirements

See CIM_LogicalDevice in the “CIM Elements” section of the *Shared Device Management Profile* for the list of mandatory properties.

6.8.3.2.3 Behavior Requirements

6.8.3.2.3.1 Preconditions

`$containerInstance` contains the instance of CIM_ComputerSystem for which the scoped CIM_LogicalDevice instances are displayed. The *Shared Device Management Profile* requires that the CIM_LogicalDevice instance be associated with its scoping system via an instance of the CIM_SystemDevice association.

`#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

6.8.3.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
   #propertylist[] = { “ElementName” }
   }
// Step 1 - find all the scoped instances
   #Error = &smOpAssociators (
      $containerInstancePath->,
      “SystemDevice”,
      #className,
      NULL,
      NULL,
      NULL,
      $devices[] );
if (0 != #Error.code)
   {
   &smProcessOpError (#Error);
   //includes &smEnd;
   }
// Step 2 - add their referenced properties
for $device in $devices[] {
   &lAddReferencedProperties ( $device, #referencedPropertyNames[] );
}
//step 3 – display them
&smShowInstancesWithReferencedProperties ( $devices[], #propertyList[],
   #referencedPropertyNames[]);
&smEnd;
```

ANNEX A
(informative)

Change Log

Version	Date	Author	Description
1.0.0	2009-06-04		DMTF Standard Release