Document Number: DSP0806

Date: 2009-06-04

Version: 1.0.0

Device Tray Profile SM CLP Command Mapping Specification

Document Type: Specification

Document Status: DMTF Standard

Document Language: E

CONTENTS

Tables

59

Foreword

The *Device Tray Profile SM CLP Command Mapping Specification* (DSP0806) 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 DTMF Server Management Working Group:

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

Introduction

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

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

Device Tray Profile SM CLP Command 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 *Device Tray 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 DSP1019, *DeviceTray Profile 1.0*,
http://www.dmtf.org/standards/published_documents/DSP1019_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?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

4.8
UFsT
User Friendly selection Tag

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 *SM CLP-to-CIM Common Mapping Specification 1.0* (DSP0216).

- smStartRSC()
- smStopRSC()
- smResetRSC()
- smShowInstance()
- smShowInstances()
- smSetInstance()
- smShowAssociationInstances()
- smShowAssociationInstance()

This mapping does not define any recipes for local reuse.

6 Mappings

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

6.1 CIM_LogicalModule

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

Table 1 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 1 is for informational purposes only; in case of a conflict between Table 1 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 1.

191 **Table 1 – Command Verb Requirements for CIM_LogicalModule**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	May	See 6.1.1.
set	May	See 6.1.2.
show	Shall	See 6.1.3.
start	May	See 6.1.4.
stop	May	See 6.1.5.

192 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, and `load`.

193 ### 6.1.1 Reset

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

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

197 #### 6.1.1.1 Reset a Single Instance

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

200 ##### 6.1.1.1.1 Command Form

201 **reset <CIM_LogicalModule *single object*>**

202 ##### 6.1.1.1.2 CIM Requirements

```
203 uint16 EnabledState;
204 uint16 RequestedState;
205 uint32 EnabledLogicalElement.RequestStateChange (
206    [IN] uint16 RequestedState = “<request value>”,
207    [OUT] REF CIM_ConcreteJob Job,
208    [IN] datetime TimeoutPeriod );
```

209 ##### 6.1.1.1.3 Behavior Requirements

```
210 $instance=<CIM_LogicalModule single object>
211 smResetRSC ( $instance.getObjectPath() );
212 &smEnd;
```

213 ### 6.1.2 Set

214 This section describes how to implement the `set` verb when it is applied to an instance of
215 CIM_LogicalModule. Implementations may support the use of the `set` verb with CIM_LogicalModule.

216 The `set` verb is used to modify descriptive properties of the CIM_LogicalModule instance.

6.1.2.1 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 *Device Tray Profile*.

6.1.2.1.1 Command Form

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

6.1.2.1.2 CIM Requirements

See CIM_LogicalModule in the “CIM Elements” section of the *Device Tray Profile* for the list of mandatory properties.

6.1.2.1.3 Behavior Requirements

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

6.1.2.2 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 *Device Tray Profile*.

6.1.2.2.1 Command Form

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

6.1.2.2.2 CIM Requirements

See CIM_LogicalModule in the “CIM Elements” section of the *Device Tray Profile* for the list of mandatory properties.

6.1.2.2.3 Behavior Requirements

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

6.1.3 Show

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

The `show` verb is used to display information about the device tray.

6.1.3.1 Show a Single Instance

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

6.1.3.1.1 Command Form

```
show <CIM_LogicalModule single object>
```

6.1.3.1.2 CIM Requirements

See CIM_LogicalModule in the “CIM Elements” section of the *Device Tray Profile* for the list of mandatory properties.

6.1.3.1.3 Behavior Requirements

6.1.3.1.3.1 Preconditions

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

6.1.3.1.3.2 Pseudo Code

```
$instance=<CIM_LogicalModule single object>
#propertylist[] = NULL;
if (false == #all)
   {
   #propertylist[] = { “ModuleNumber”, “EnabledState”, “RequestedState”,
      “EnabledDefault”, “OperationalStatus”, “StatusDescriptions”,
      “LogicalModuleType” }
   }
&smShowInstance ( $instance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.1.3.2 Show Multiple Instances

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

6.1.3.2.1 Command Form

```
show <CIM_LogicalModule multiple objects>
```

6.1.3.2.2 CIM Requirements

See CIM_LogicalModule in the “CIM Elements” section of the *Device Tray Profile* for the list of mandatory properties.

6.1.3.2.3 Behavior Requirements

6.1.3.2.3.1 Preconditions

`$containerInstance` contains the instance of CIM_ComputerSystem for which scoped device trays (CIM_LogicalModule instances) are being displayed. The *Device Tray Profile* requires that the CIM_LogicalModule 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.

6.1.3.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if (false == #all)
   {
   #propertylist[] = {"ModuleNumber", "EnabledState", "RequestedState",
      "EnabledDefault", "OperationalStatus", "StatusDescriptions",
      "LogicalModuleType" }
   }
&smShowInstances ( "CIM_LogicalModule", "CIM_SystemDevice",
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.1.4 Start

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

The `start` verb is used to enable a device tray.

6.1.4.1 Start a Single Instance

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

6.1.4.1.1 Command Form

```
start <CIM_LogicalModule single object>
```

6.1.4.1.2 CIM Requirements

```
uint16 EnabledState;
uint16 RequestedState;
uint32 EnabledLogicalElement.RequestStateChange (
   [IN] uint16 RequestedState = "<request value>",
   [OUT] REF CIM_ConcreteJob Job,
   [IN] datetime TimeoutPeriod );
```

6.1.4.1.3 Behavior Requirements

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

6.1.5 Stop

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

The `stop` verb is used to disable a device tray.

6.1.5.1 Stop a Single Instance

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

6.1.5.1.1 Command Form

```
stop <CIM_LogicalModule single object>
```

6.1.5.1.2 CIM Requirements

```
uint16 EnabledState;
uint16 RequestedState;
uint32 EnabledLogicalElement.RequestStateChange (
   [IN] uint16 RequestedState = “<request value>”,
   [OUT] REF CIM_ConcreteJob Job,
   [IN] datetime TimeoutPeriod );
```

6.1.5.1.3 Behavior Requirements

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

6.2 CIM_SystemDevice

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 instances of 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_SystemDevice

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 6.2.1.
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`.

6.2.1 Show

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

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

6.2.1.1 Show Multiple Instances

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

6.2.1.1.1 Command Form

```
show <CIM_SystemDevice multiple objects>
```

6.2.1.1.2 CIM Requirements

See CIM_SystemDevice in the "CIM Elements" section of the *Device Tray Profile* for the list of mandatory properties.

6.2.1.1.2.1 Behavior Requirements

6.2.1.1.2.2 Preconditions

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

6.2.1.1.2.3 Pseudo Code

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

6.2.1.2 Show a Single Instance – CIM_LogicalModule 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_SystemDevice where the reference specified is to an instance of CIM_LogicalModule. An instance of CIM_LogicalModule is referenced by exactly one instance of CIM_SystemDevice. Therefore, a single instance will be returned.

6.2.1.2.1 Command Form

```
show <CIM_SystemDevice single object>
```

6.2.1.2.2 CIM Requirements

See CIM_SystemDevice in the "CIM Elements" section of the *Device Tray Profile* for the list of mandatory properties.

6.2.1.2.3 Behavior Requirements

6.2.1.2.3.1 Preconditions

`$instance` contains the instance of CIM_LogicalModule which is referenced by CIM_SystemDevice.

6.2.1.2.3.2 Pseudo Code

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

6.2.1.3 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_SystemDevice where both references are specified and therefore the desired instance is unambiguously identified.

6.2.1.3.1 Command Form

```
show <CIM_SystemDevice single object>
```

6.2.1.3.2 CIM Requirements

See CIM_SystemDevice in the “CIM Elements” section of the *Device Tray Profile* for the list of mandatory properties.

6.2.1.3.3 Behavior Requirements

6.2.1.3.3.1 Preconditions

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

`$instanceB` contains the instance of CIM_LogicalModule which is referenced by CIM_SystemDevice.

6.2.1.3.3.2 Pseudo Code

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

6.3 CIM_ConcreteComponent

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_ConcreteComponent

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 6.3.1.
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 Show

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

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

6.3.1.1 Show Multiple Instances

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

6.3.1.1.1 Command Form

```
show <CIM_ConcreteComponent multiple objects>
```

6.3.1.1.2 CIM Requirements

See CIM_ConcreteComponent in the “CIM Elements” section of the *Device Tray Profile* for the list of mandatory properties.

6.3.1.1.3 Behavior Requirements

6.3.1.1.3.1 Preconditions

`$instance` contains the instance of CIM_LogicalModule which is referenced by CIM_ConcreteComponent.

6.3.1.1.3.2 Pseudo Code

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

6.3.1.2 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_ConcreteComponent where the reference specified is to an instance of CIM_LogicalDevice. An instance of CIM_LogicalDevice is referenced by exactly one instance of CIM_ConcreteComponent. Therefore, a single instance will be returned.

6.3.1.2.1 Command Form

```
show <CIM_ConcreteComponent single object>
```

6.3.1.2.2 CIM Requirements

See CIM_ConcreteComponent in the “CIM Elements” section of the *Device Tray Profile* for the list of mandatory properties.

6.3.1.2.3 Behavior Requirements

6.3.1.2.3.1 Preconditions

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

6.3.1.2.3.2 Pseudo Code

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

6.3.1.3 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_ConcreteComponent where both references are specified and therefore the desired instance is unambiguously identified.

6.3.1.3.1 Command Form

```
show <CIM_ConcreteComponent single object>
```

6.3.1.3.2 CIM Requirements

See CIM_ConcreteComponent in the “CIM Elements” section of the *Device Tray Profile* for the list of mandatory properties.

6.3.1.3.3 Behavior Requirements

6.3.1.3.3.1 Preconditions

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

`$instanceB` contains the instance of CIM_LogicalModule which is referenced by CIM_ConcreteComponent.

6.3.1.3.3.2 Pseudo Code

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

6.4 CIM_ElementCapabilities

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 instances of 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_ElementCapabilities

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

Command Verb	Requirement	Comments
show	Shall	See 6.4.1.
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 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.

6.4.1.1 Show Multiple Instances

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 only one reference is specified and the reference is to an instance of CIM_LogicalModule.

6.4.1.1.1 Command Form

```
show <CIM_ElementCapabilities multiple objects>
```

6.4.1.1.2 CIM Requirements

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

6.4.1.1.3 Behavior Requirements

6.4.1.1.3.1 Preconditions

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

6.4.1.1.3.2 Pseudo Code

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

6.4.1.2 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_LogicalModule. An instance of CIM_LogicalModule is referenced by exactly one instance of CIM_ElementCapabilities. Therefore, a single instance will be returned.

6.4.1.2.1 Command Form

```
show <CIM_ElementCapabilities single object>
```

6.4.1.2.2 CIM Requirements

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

6.4.1.2.3 Behavior Requirements

6.4.1.2.3.1 Preconditions

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

6.4.1.2.3.2 Pseudo Code

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

6.4.1.3 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_ElementCapabilities where both references are specified and therefore the desired instance is unambiguously identified.

6.4.1.3.1 Command Form

```
show <CIM_ElementCapabilities single object>
```

6.4.1.3.2 CIM Requirements

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

6.4.1.3.3 Behavior Requirements

6.4.1.3.3.1 Preconditions

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

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

6.4.1.3.3.2 Pseudo Code

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

6.5 CIM_EnabledLogicalElementCapabilities

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 instances of 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.

551 **Table 5 – 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	Not supported	
show	Shall	See 6.5.1.
start	Not supported	
stop	Not supported	

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

554 ### 6.5.1 Show

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

558 The `show` verb is used to display information about the device tray.

559 #### 6.5.1.1 Show a Single Instance

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

562 ##### 6.5.1.1.1 Command Form

563 **`show <CIM_EnabledLogicalElementCapabilities`** ***`single object`*****`>`**

564 ##### 6.5.1.1.2 CIM Requirements

565 See CIM_EnabledLogicalElementCapabilities in the “CIM Elements” section of the *Device Tray Profile* for
566 the list of mandatory properties.

567 ##### 6.5.1.1.3 Behavior Requirements

568 ###### 6.5.1.1.3.1 Preconditions

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

570 ###### 6.5.1.1.3.2 Pseudo Code

```
571 $instance=<CIM_EnabledLogicalElementCapabilities single object>
572 if (#all)
573    {
574    #propertylist[] = { “InstanceID”, “RequestedStatesSupported”,
575       “ElementNameEditSupported”, “MaxElementNameLen” }
576    }
```

```
else
   {
   #propertylist[] = { “RequestedStatesSupported”, “ElementNameEditSupported”,
      “MaxElementNameLen” }
   }
&smShowInstance ( $instance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.5.1.2 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.

6.5.1.2.1 Command Form

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

6.5.1.2.2 CIM Requirements

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

6.5.1.2.3 Behavior Requirements

6.5.1.2.3.1 Preconditions

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

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

6.5.1.2.3.2 Pseudo Code

```
if (#all)
   {
   #propertylist[] = { “InstanceID”, “RequestedStatesSupported”,
      “ElementNameEditSupported”, “MaxElementNameLen” }
   }
else
   {
   #propertylist[] = { “RequestedStatesSupported”, “ElementNameEditSupported”,
      “MaxElementNameLen” }
   }
&smShowInstances ( “CIM_EnabledLogicalElementCapabilities”, “CIM_MemberOfCollection”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

ANNEX A
(informative)

Change Log

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