Document Number: DSP0812

Date: 2009-07-14

Version: 1.0.0

Physical Asset Profile SM CLP Mapping Specification

Document Type: Specification

Document Status: DMTF Standard

Document Language: E

CONTENTS

Tables

Foreword

The *Physical Asset Profile SM CLP Mapping Specification* (DSP0812) 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:

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

Introduction

This document defines the SM CLP mapping for CIM elements described in the *Physical Asset 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 *Physical Asset Profile* using CIM operations.

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

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

DMTF DSP1011, *Physical Asset Profile 1.0*,
http://www.dmtf.org/standards/published_documents/DSP1011_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

Storage Management Initiative

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.

- smShowInstance
- smShowInstances
- smShowAssociationInstance
- smShowAssociationInstances

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 *Physical Asset Profile*.

6.1 CIM_Card

The `cd`, `exit`, `help`, and `version` 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 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.

Table 1 – Command Verb Requirements for CIM_Card

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 6.1.2.
Start	Not supported	
Stop	Not supported	

No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `exit`, `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_Card 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_Card. Implementations shall support the use of the `show` verb with CIM_Card.

6.1.2.1 Show Command Form for Multiple Instances Target – CIM_PhysicalPackage Container Instance

This command form is used to show many instances of CIM_Card when CIM_PhysicalPackage is the container instance.

6.1.2.2 Command Form

```
show <CIM_Card multiple instances>
```

6.1.2.3 CIM Requirements

See CIM_Card in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.1.2.4 Behavior Requirements

6.1.2.4.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalPackage and is associated to the targeted instances of CIM_Card through the CIM_Container association.

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

6.1.2.4.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_Card”, “CIM_Container”, $containerInstance.getInstancePath(),
   #propertylist[] );
&smEnd;
```

6.1.2.5 Show Command Form for Multiple Instances Target – CIM_PhysicalConnector Container Instance

This command form is used to show many instances of CIM_Card when CIM_PhysicalConnector is the container instance.

6.1.2.5.1 Command Form

```
show <CIM_Card multiple instances>
```

6.1.2.5.2 CIM Requirements

See CIM_Card in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.1.2.5.3 Behavior Requirements

6.1.2.5.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalConnector and is associated to the targeted instances of CIM_Card through the CIM_PackageInConnector association.

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

6.1.2.5.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_Card”, “CIM_PackageInConnector”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.1.2.6 Show Command Form for Multiple Instances Target – CIM_ConcreteCollection Container Instance

This command form is used to show many instances of CIM_Card when CIM_ConcreteCollection is the container instance.

6.1.2.6.1 Command Form

```
show <CIM_Card multiple instances>
```

6.1.2.6.2 CIM Requirements

See CIM_Card in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.1.2.6.3 Behavior Requirements

6.1.2.6.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_ConcreteCollection and is associated to the targeted instances of CIM_Card through the CIM_MemberOfCollection association.

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

6.1.2.6.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_Card”, “CIM_MemberOfCollection”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.1.2.7 Show Command Form for a Single Instance Target

This command form is used to show a single instance of CIM_Card.

6.1.2.7.1 Command Form

```
show <CIM_Card single instance>
```

6.1.2.7.2 CIM Requirements

See CIM_Card in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.1.2.7.3 Behavior Requirements

6.1.2.7.3.1 Preconditions

In this section `$instance` represents the targeted instance of CIM_Card.

```
$instance=<CIM_Card single instance>;
```

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

6.1.2.7.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstance ( $instance, #propertylist[] );
&smEnd;
```

6.2 CIM_Chassis

The `cd`, `exit`, `help`, and `version` 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_Chassis

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 6.2.2.
Start	Not supported	
Stop	Not supported	

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

6.2.1 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_Chassis are unordered; therefore, no algorithm is defined.

6.2.2 Show

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

6.2.2.1 Show Command Form for Multiple Instances Target – CIM_PhysicalPackage Container Instance

This command form is used to show many instances of CIM_Chassis when CIM_PhysicalPackage is the container instance.

6.2.2.1.1 Command Form

```
show <CIM_Chassis multiple instances>
```

6.2.2.1.2 CIM Requirements

See CIM_Chassis in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties.

6.2.2.1.3 Behavior Requirements

6.2.2.1.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalPackage and is associated to the targeted instances of CIM_Chassis through the CIM_Container association.

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

6.2.2.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_Chassis”, “CIM_Container”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.2.2.2 Show Command Form for Multiple Instances Target – CIM_PhysicalConnector Container Instance

This command form is used to show many instances of CIM_Chassis when CIM_PhysicalConnector is the container instance.

6.2.2.2.1 Command Form

```
show <CIM_Chassis multiple instances>
```

6.2.2.2.2 CIM Requirements

See CIM_Chassis in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.2.2.2.3 Behavior Requirements

6.2.2.2.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalConnector and is associated to the targeted instances of CIM_Chassis through the CIM_PackageInConnector association.

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

6.2.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_Chassis”, “CIM_PackageInConnector”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.2.2.3 Show Command Form for Multiple Instances Target – CIM_ConcreteCollection Container Instance

This command form is used to show many instances of CIM_Chassis when CIM_ConcreteCollection is the container instance.

6.2.2.3.1 Command Form

```
show <CIM_Chassis multiple instances>
```

6.2.2.3.2 CIM Requirements

See CIM_Chassis in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.2.2.3.3 Behavior Requirements

6.2.2.3.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_ConcreteCollection and is associated to the targeted instances of CIM_Chassis through the CIM_MemberOfCollection association.

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

6.2.2.3.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_Chassis”, “CIM_MemberOfCollection”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.2.2.4 Show Command Form for a Single Instance Target

This command form is used to show a single instance of CIM_Chassis.

6.2.2.4.1 Command Form

`show <CIM_Chassis` ***`single instance`*** **`>`**

6.2.2.4.2 CIM Requirements

See CIM_Chassis in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.2.2.4.3 Behavior Requirements

6.2.2.4.3.1 Preconditions

In this section `$instance` represents the targeted instance of CIM_Chassis.

`$instance=<CIM_Chassis` *`single instance`*`>;`

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

6.2.2.4.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstance ( $instance, #propertylist[] );
&smEnd;
```

6.3 CIM_Chip

The `cd`, `exit`, `help`, and `version` 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 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_Chip

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 6.3.2.
Start	Not supported	
Stop	Not supported	

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

6.3.1 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_Chip 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_Chip. Implementations shall support the use of the `show` verb with CIM_Chip.

6.3.2.1 Show Command Form for Multiple Instances Target – CIM_PhysicalPackage Container Instance

This command form is used to show many instances of CIM_Chip when CIM_PhysicalPackage is the container instance.

6.3.2.1.1 Command Form

```
show <CIM_Chip multiple instances>
```

6.3.2.1.2 CIM Requirements

See CIM_Chip in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.3.2.1.3 Behavior Requirements

6.3.2.1.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalPackage and is associated to the targeted instances of CIM_Chip through the CIM_Container association.

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

6.3.2.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_Chip”, “CIM_Container”, $containerInstance.getInstancePath(),
   #propertylist[] );
&smEnd;
```

6.3.2.2 Show Command Form for Multiple Instances Target – CIM_ConcreteCollection Container Instance

This command form is used to show many instances of CIM_Chip when CIM_ConcreteCollection is the container instance.

6.3.2.2.1 Command Form

```
show <CIM_Chip multiple instances>
```

6.3.2.2.2 CIM Requirements

See CIM_Chip in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.3.2.2.3 Behavior Requirements

6.3.2.2.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_ConcreteCollection and is associated to the targeted instances of CIM_Chip through the CIM_MemberOfCollection association.

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

6.3.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_Chip”, “CIM_MemberOfCollection”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.3.2.3 Show Command Form for a Single Instance Target

This command form is used to show a single instance of CIM_Chip.

6.3.2.3.1 Command Form

```
show <CIM_Chip single instance>
```

6.3.2.3.2 CIM Requirements

See CIM_Chip in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.3.2.3.3 Behavior Requirements

6.3.2.3.3.1 Preconditions

In this section `$instance` represents the targeted instance of CIM_Chip.

```
$instance=<CIM_Chip single instance>;
```

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

6.3.2.3.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstance ( $instance, #propertylist[] );
&smEnd;
```

6.4 CIM_ComputerSystemPackage

The `cd`, `exit`, `help`, and `version` 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_ComputerSystemPackage

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 mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `exit`, `load`, `reset`, `set`, `start`, and `stop`.

6.4.1 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_ComputerSystemPackage 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_ComputerSystemPackage. Implementations shall support the use of the `show` verb with CIM_ComputerSystemPackage.

6.4.2.1 Show Command Form for Multiple Instances Target – CIM_ComputerSystem

This command form is used to show many instances of CIM_ComputerSystemPackage. This command form corresponds to a `show` command issued against CIM_ComputerSystemPackage where only one reference is specified and the reference is to an instance of CIM_ComputerSystem.

6.4.2.1.1 Command Form

```
show <CIM_ComputerSystemPackage multiple instances>
```

6.4.2.1.2 CIM Requirements

See CIM_ComputerSystemPackage in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.4.2.1.3 Behavior Requirements

6.4.2.1.3.1 Preconditions

In this section `$instance` represents the instance of CIM_ComputerSystem which is referenced by CIM_ComputerSystemPackage.

Specifying the “-all” option does not change the output because the only property on the target instance is mandatory in “CIM Elements” section of the *Physical Asset Profile*. Thus, no additional pseudo code is required to handle the option.

6.4.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ComputerSystemPackage”,
    $instance.getInstancePath() );
&smEnd;
```

6.4.2.2 Show Command Form for Multiple Instances – CIM_PhysicalPackage Reference

This command form is used to show multiple instances of CIM_ComputerSystemPackage. This command form corresponds to a `show` command issued against multiple instances of CIM_ComputerSystemPackage where only one reference is specified and the reference is to an instance of CIM_PhysicalPackage or a subclass of CIM_PhysicalPackage.

6.4.2.2.1 Command Form

```
show <CIM_ComputerSystemPackage multiple instances>
```

6.4.2.2.2 CIM Requirements

See CIM_ComputerSystemPackage in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.4.2.2.3 Behavior Requirements

6.4.2.2.3.1 Preconditions

In this section `$instance` represents the instance of CIM_PhysicalPackage or a subclass of CIM_PhysicalPackage which is referenced by CIM_ComputerSystemPackage.

Specifying the "-all" option does not change the output since the only property on the target instance is mandatory in the "CIM Elements" section of the *Physical Asset Profile*. Thus, no additional pseudo code is required to handle the option.

6.4.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( "CIM_ComputerSystemPackage",
   $instance.getInstancePath() );
&smEnd;
```

6.4.2.3 Show Command Form for 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_ComputerSystemPackage where both references are specified and therefore the desired instance is unambiguously identified.

6.4.2.3.1 Command Form

```
show <CIM_ComputerSystemPackage single instance>
```

6.4.2.3.2 CIM Requirements

See CIM_ComputerSystemPackage in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.4.2.3.3 Behavior Requirements

6.4.2.3.3.1 Preconditions

In this section `$instanceA` represents the referenced instance of CIM_ComputerSystem through CIM_ComputerSystemPackage association. `$instanceB` represents the instance of CIM_PhysicalPackage or a subclass of CIM_PhysicalPackage which is referenced by CIM_ComputerSystemPackage.

Specifying the "-all" option does not change the output since the only property on the target instance is mandatory in the "CIM Elements" section of the *Physical Asset Profile*. Thus, no additional pseudo code is required to handle the option.

6.4.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( "CIM_ComputerSystemPackage",
   $instanceA.getInstancePath(),$instanceB.getInstancePath() );
&smEnd;
```

6.5 CIM_ConfigurationCapacity

The `cd`, `exit`, `help`, and `version` 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_ConfigurationCapacity

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 6.5.2.
Start	Not supported	
Stop	Not supported	

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

6.5.1 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_ConfigurationCapacity are unordered; therefore, no algorithm is defined.

6.5.2 Show

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

6.5.2.1 Show Command Form for Multiple Instances Target

This command form is used to show many instances of CIM_ConfigurationCapacity.

6.5.2.1.1 Command Form

```
show <CIM_ConfigurationCapacity multiple instances>
```

6.5.2.1.2 CIM Requirements

See CIM_ConfigurationCapacity in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.5.2.1.3 Behavior Requirements

6.5.2.1.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_ConcreteCollection, and is associated to the targeted instances of CIM_ConfigurationCapacity through the CIM_MemberOfCollection association.

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

6.5.2.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_ConfigurationCapacity”, “CIM_MemberOfCollection”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.5.2.2 Show Command Form for a Single Instance Target

This command form is used to show a single instance of CIM_ConfigurationCapacity.

6.5.2.2.1 Command Form

`show <CIM_ConfigurationCapacity` *`single instance`*`>`

6.5.2.2.2 CIM Requirements

See CIM_ConfigurationCapacity in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.5.2.2.3 Behavior Requirements

6.5.2.2.3.1 Preconditions

In this section `$instance` represents the targeted instance of CIM_ConfigurationCapacity.

`$instance=<CIM_ConfigurationCapacity` *`single instance`*`>;`

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

6.5.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstance ( $instance, #propertylist[] );
&smEnd;
```

6.6 CIM_ConnectedTo

The `cd`, `exit`, `help`, and `version` 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_ConnectedTo

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 6.6.2.
Start	Not supported	
Stop	Not supported	

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

6.6.1 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_ConnectedTo 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_ConnectedTo. Implementations shall support the use of the `show` verb with CIM_ConnectedTo.

6.6.2.1 Show Command Form for Multiple Instances Target

This command form is used to show many instances of CIM_ConnectedTo. This command form corresponds to a `show` command issued against instances of CIM_ConnectedTo where only one reference is specified and the reference is to the instance of CIM_PhysicalConnector or CIM_Slot.

6.6.2.1.1 Command Form

```
show <CIM_ConnectedTo multiple instances>
```

6.6.2.1.2 CIM Requirements

See CIM_ConnectedTo in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.6.2.1.3 Behavior Requirements

6.6.2.1.3.1 Preconditions

In this section `$instance` represents the instance of CIM_PhysicalConnector or CIM_Slot which is referenced by CIM_ConnectedTo.

6.6.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ConnectedTo”, $instance.getInstancePath() );
&smEnd;
```

6.6.2.2 Show Command Form for 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_ConnectedTo where both references are specified and therefore the desired instance is unambiguously identified.

6.6.2.2.1 Command Form

```
show <CIM_ConnectedTo single instance>
```

6.6.2.2.2 CIM Requirements

See CIM_ConnectedTo in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.6.2.2.3 Behavior Requirements

6.6.2.2.3.1 Preconditions

In this section `$instanceA` represents the referenced instance of CIM_PhysicalConnector or CIM_Slot through CIM_ConnectedTo association. `$instanceB` represents the other instance of CIM_PhysicalConnector or CIM_Slot which is referenced by CIM_ConnectedTo.

6.6.2.2.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_ConnectedTo”, $instanceA.getInstancePath(),
   $instanceB.getInstancePath() );
&smEnd;
```

6.7 CIM_Container

The `cd`, `exit`, `help`, and `version` 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_Container

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 6.7.2.
Start	Not supported	
Stop	Not supported	

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

6.7.1 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_Container are unordered; therefore, no algorithm is defined.

6.7.2 Show

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

6.7.2.1 Show Command Form for Multiple Instances Target – CIM_PhysicalPackage Reference

This command form is used to show many instances of CIM_Container. This command form corresponds to a `show` command issued against instances of CIM_Container where only one reference is specified and the reference is to the instance of CIM_PhysicalPackage or a subclass of CIM_PhysicalPackage.

6.7.2.1.1 Command Form

```
show <CIM_Container multiple instances>
```

6.7.2.1.2 CIM Requirements

See CIM_Container in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.7.2.1.3 Behavior Requirements

6.7.2.1.3.1 Preconditions

In this section `$instance` represents the instance of CIM_PhysicalPackage or a subclass of CIM_PhysicalPackage which is referenced by CIM_Container.

6.7.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_Container”, $instance.getInstancePath() );
&smEnd;
```

6.7.2.2 Show Command Form for a Single Instance – CIM_PhysicalElement Reference

This command form is used to show a single instance of CIM_Container. This command form corresponds to a `show` command issued against a single instance of CIM_Container where only one reference is specified and the reference is to the instance of a subclass of CIM_PhysicalElement.

6.7.2.2.1 Command Form

```
show <CIM_Container single instance>
```

6.7.2.2.2 CIM Requirements

See CIM_Container in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.7.2.2.3 Behavior Requirements

6.7.2.2.3.1 Preconditions

In this section `$instance` represents the instance of a subclass of CIM_PhysicalElement which is referenced by CIM_Container.

6.7.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_Container”, $instance.getInstancePath() );
&smEnd;
```

6.7.2.3 Show Command Form for 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_Container where both references are specified and therefore the desired instance is unambiguously identified.

6.7.2.3.1 Command Form

```
show <CIM_Container single instance>
```

6.7.2.3.2 CIM Requirements

See CIM_Container in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.7.2.3.3 Behavior Requirements

6.7.2.3.3.1 Preconditions

In this section `$instanceA` represents the referenced instance of a subclass of CIM_PhysicalElement through CIM_Container association. `$instanceB` represents the instance of CIM_PhysicalPackage or a subclass of CIM_PhysicalPackage which is referenced by CIM_Container.

6.7.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_Container”, $instanceA.getInstancePath(),
   $instanceB.getInstancePath() );
&smEnd;
```

6.8 CIM_ElementCapabilities

The `cd`, `exit`, `help`, and `version` 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_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 6.8.2.
Start	Not supported	
Stop	Not supported	

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

6.8.1 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.

6.8.2 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.

6.8.2.1 Show Command Form for a Single Instance Target – CIM_PhysicalElement Reference

This command form is used to show a single instance of CIM_ElementCapabilities. This command form corresponds to a `show` command issued against a single instance of CIM_ElementCapabilities where only one reference is specified and the reference is to the instance of a subclass of CIM_PhysicalElement.

6.8.2.1.1 Command Form

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

6.8.2.1.2 CIM Requirements

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

6.8.2.1.3 Behavior Requirements

6.8.2.1.3.1 Preconditions

In this section `$instance` represents the instance of a subclass of CIM_PhysicalElement which is referenced by CIM_ElementCapabilities.

6.8.2.1.3.2 Pseudo Code

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

6.8.2.2 Show Command Form for Multiple Instances –CIM_PhysicalAssetCapabilities Reference

This command form is used to show multiple instances of CIM_ElementCapabilities. This command form corresponds to a `show` command issued against multiple instances of CIM_ElementCapabilities where only one reference is specified and the reference is to the instance of CIM_PhysicalAssetCapabilities.

6.8.2.2.1 Command Form

```
show <CIM_ElementCapabilities multiple instances>
```

6.8.2.2.2 CIM Requirements

See CIM_ElementCapabilities in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.8.2.2.3 Behavior Requirements

6.8.2.2.3.1 Preconditions

In this section `$instance` represents the instance of CIM_PhysicalAssetCapabilities which is referenced by CIM_ElementCapabilities.

6.8.2.2.3.2 Pseudo Code

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

6.8.2.3 Show Command Form for 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.

6.8.2.3.1 Command Form

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

6.8.2.3.2 CIM Requirements

See CIM_ElementCapabilities in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.8.2.3.3 Behavior Requirements

6.8.2.3.3.1 Preconditions

In this section `$instanceA` represents the referenced instance of a subclass of CIM_PhysicalElement through CIM_ElementCapabilities association. `$instanceB` represents the instance of CIM_PhysicalAssetCapabilities which is referenced by CIM_ElementCapabilities.

6.8.2.3.3.2 Pseudo Code

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

6.9 CIM_ElementCapacity

The `cd`, `exit`, `help`, and `version` 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_ElementCapacity

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 6.9.2.
Start	Not supported	
Stop	Not supported	

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

6.9.1 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_ElementCapacity are unordered; therefore, no algorithm is defined.

6.9.2 Show

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

6.9.2.1 Show Command Form for Multiple Instances Target

This command form is used to show many instances of CIM_ElementCapacity. This command form corresponds to a `show` command issued against instances of CIM_ElementCapacity where only one reference is specified and the reference is to the instance of a subclass of CIM_PhysicalElement.

6.9.2.1.1 Command Form

```
show <CIM_ElementCapacity multiple instances>
```

6.9.2.1.2 CIM Requirements

See CIM_ElementCapacity in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.9.2.1.3 Behavior Requirements

6.9.2.1.3.1 Preconditions

In this section `$instance` represents the instance of a subclass of CIM_PhysicalElement which is referenced by CIM_ElementCapacity.

6.9.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ElementCapacity”, $instance.getInstancePath() );
&smEnd;
```

6.9.2.2 Show Command Form for Multiple Instances – CIM_ConfigurationCapacity Reference

This command form is used to show multiple instances of CIM_ElementCapacity. This command form corresponds to a `show` command issued against multiple instances of CIM_ElementCapacity where only one reference is specified and the reference is to the instance of CIM_PhysicalAssetCapabilities.

6.9.2.2.1 Command Form

```
show <CIM_ElementCapacity multiple instances>
```

6.9.2.2.2 CIM Requirements

See CIM_ElementCapacity in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.9.2.2.3 Behavior Requirements

6.9.2.2.3.1 Preconditions

In this section `$instance` represents the instance of CIM_ConfigurationCapacity which is referenced by CIM_ElementCapacity.

6.9.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ElementCapacity”, $instance.getInstancePath() );
&smEnd;
```

6.9.2.3 Show Command Form for 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_ElementCapacity where both references are specified and therefore the desired instance is unambiguously identified.

6.9.2.3.1 Command Form

```
show <CIM_ElementCapacity single instance>
```

6.9.2.3.2 CIM Requirements

See CIM_ElementCapacity in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.9.2.3.3 Behavior Requirements

6.9.2.3.3.1 Preconditions

In this section `$instanceA` represents the referenced instance of a subclass of CIM_PhysicalElement through CIM_ElementCapacity association. `$instanceB` represents the instance of CIM_ConfigurationCapacity which is referenced by CIM_ElementCapacity.

6.9.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( "CIM_ElementCapacity", $instanceA.getInstancePath(),
   $instanceB.getInstancePath() );
&smEnd;
```

6.10 CIM_PackageInConnector

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

Table 10 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 10 is for informational purposes only; in case of a conflict between Table 10 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 10.

Table 10 – Command Verb Requirements for CIM_PackageInConnector

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 6.10.2.
Start	Not supported	
Stop	Not supported	

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

6.10.1 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_PackageInConnector are unordered; therefore, no algorithm is defined.

6.10.2 Show

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

6.10.2.1 Show Command Form for Multiple Instances Target – CIM_PhysicalPackage Reference

This command form is used to show many instances of CIM_PackageInConnector. This command form corresponds to a `show` command issued against instances of CIM_PackageInConnector where only one reference is specified and the reference is to the instance of CIM_PhysicalPackage or a subclass of CIM_PhysicalPackage.

6.10.2.1.1 Command Form

```
show <CIM_PackageInConnector multiple instances>
```

6.10.2.1.2 CIM Requirements

See CIM_PackageInConnector in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.10.2.1.3 Behavior Requirements

6.10.2.1.3.1 Preconditions

In this section `$instance` represents the instance of CIM_PhysicalPackage or a subclass of CIM_PhysicalPackage which is referenced by CIM_PackageInConnector.

6.10.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_PackageInConnector”, $instance.getInstancePath() );
&smEnd;
```

6.10.2.2 Show Command Form for Multiple Instances – CIM_PhysicalConnector Reference

This command form is used to show multiple instances of CIM_PackageInConnector. This command form corresponds to a `show` command issued against multiple instances of CIM_PackageInConnector where only one reference is specified and the reference is to the instance of CIM_PhysicalConnector.

6.10.2.2.1 Command Form

```
show <CIM_PackageInConnector multiple instances>
```

6.10.2.2.2 CIM Requirements

See CIM_PackageInConnector in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.10.2.2.3 Behavior Requirements

6.10.2.2.3.1 Preconditions

In this section `$instance` represents the instance of CIM_PhysicalConnector which is referenced by CIM_PackageInConnector.

6.10.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_PackageInConnector”, $instance.getInstancePath() );
&smEnd;
```

6.10.2.3 Show Command Form for a Single Instance – Both References

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

6.10.2.3.1 Command Form

```
show <CIM_PackageInConnector single instance>
```

6.10.2.3.2 CIM Requirements

See CIM_PackageInConnector in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.10.2.3.3 Behavior Requirements

6.10.2.3.3.1 Preconditions

In this section `$instanceA` represents the referenced instance of CIM_PhysicalPackage or a subclass of CIM_PhysicalPackage through CIM_PackageInConnector association. `$instanceB` represents the instance of CIM_PhysicalConnector which is referenced by CIM_PackageInConnector.

6.10.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_PackageInConnector”, $instanceA.getInstancePath(),
   $instanceB.getInstancePath() );
&smEnd;
```

6.11 CIM_PhysicalAssetCapabilities

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

Table 11 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 11 is for informational purposes only; in case of a conflict between Table 11 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 11.

Table 11 – Command Verb Requirements for CIM_PhysicalAssetCapabilities

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 6.11.2.
Start	Not supported	
Stop	Not supported	

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

6.11.1 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_PhysicalAssetCapabilities are unordered; therefore, no algorithm is defined.

6.11.2 Show

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

6.11.2.1 Show Command Form for Multiple Instances Target

This command form is used to show many instances of CIM_PhysicalAssetCapabilities.

6.11.2.1.1 Command Form

```
show <CIM_PhysicalAssetCapabilities multiple instances>
```

6.11.2.1.2 CIM Requirements

See CIM_PhysicalAssetCapabilities in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.11.2.1.3 Behavior Requirements

6.11.2.1.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_ConcreteCollection, and is associated to the targeted instances of CIM_PhysicalAssetCapabilities through the CIM_MemberOfCollection association.

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

6.11.2.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_PhysicalAssetCapabilities”, “CIM_MemberOfCollection”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.11.2.2 Show Command Form for a Single Instance Target

This command form is used to show a single instance of CIM_PhysicalAssetCapabilities.

6.11.2.2.1 Command Form

```
show <CIM_PhysicalAssetCapabilities single instance>
```

6.11.2.2.2 CIM Requirements

See CIM_PhysicalAssetCapabilities in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.11.2.2.3 Behavior Requirements

6.11.2.2.3.1 Preconditions

In this section `$instance` represents the targeted instance of CIM_PhysicalAssetCapabilities.

```
$instance=<CIM_PhysicalAssetCapabilities single instance>;
```

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

6.11.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstance ( $instance, #propertylist[] );
&smEnd;
```

6.12 CIM_PhysicalComponent

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

Table 12 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 12 is for informational purposes only; in case of a conflict between Table 12 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 12.

Table 12 – Command Verb Requirements for CIM_PhysicalComponent

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 6.12.2.
Start	Not supported	
Stop	Not supported	

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

6.12.1 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_PhysicalComponent are unordered; therefore, no algorithm is defined.

6.12.2 Show

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

6.12.2.1 Show Command Form for Multiple Instances Target – CIM_PhysicalPackage Container Instance

This command form is used to show many instances of CIM_PhysicalComponent when CIM_PhysicalPackage is the container instance.

6.12.2.1.1 Command Form

```
show <CIM_PhysicalComponent multiple instances>
```

6.12.2.1.2 CIM Requirements

See the “CIM Elements” section in the *Physical Asset Profile*.

6.12.2.1.3 Behavior Requirements

6.12.2.1.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalPacakge and is associated to the targeted instances of CIM_PhysicalComponent through the CIM_Container association.

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

6.12.2.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( "CIM_PhysicalComponent", "CIM_Container",
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.12.2.2 Show Command Form for Multiple Instances Target – CIM_ConcreteCollection Container Instance

This command form is used to show many instances of CIM_PhysicalComponent when CIM_ConcreteCollection is the container instance.

6.12.2.2.1 Command Form

```
show <CIM_PhysicalComponent multiple instances>
```

6.12.2.2.2 CIM Requirements

See the "CIM Elements" section in the *Physical Asset Profile*.

6.12.2.2.3 Behavior Requirements

6.12.2.2.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_ConcreteCollection and is associated to the targeted instances of CIM_PhysicalComponent through the CIM_MemberOfCollection association.

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

6.12.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( "CIM_PhysicalComponent", "CIM_MemberOfCollection",
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.12.2.3 Show Command Form for a Single Instance Target

This command form is used to show a single instance of CIM_PhysicalComponent.

6.12.2.3.1 Command Form

```
show <CIM_PhysicalComponent single instance>
```

6.12.2.3.2 CIM Requirements

See the "CIM Elements" section in the *Physical Asset Profile*.

6.12.2.3.3 Behavior Requirements

6.12.2.3.3.1 Preconditions

In this section `$instance` represents the targeted instance of CIM_PhysicalComponent.

```
$instance=<CIM_PhysicalComponent single instance>;
```

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

6.12.2.3.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstance ( $instance, #propertylist[] );
&smEnd;
```

6.13 CIM_PhysicalConnector

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

Table 13 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 13 is for informational purposes only; in case of a conflict between Table 13 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 13.

Table 13 – Command Verb Requirements for CIM_PhysicalConnector

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 6.13.2.
Start	Not supported	
Stop	Not supported	

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

6.13.1 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_PhysicalConnector are unordered; therefore, no algorithm is defined.

6.13.2 Show

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

6.13.2.1 Show Command Form for Multiple Instances Target – CIM_PhysicalPackage Container Instance

This command form is used to show many instances of CIM_PhysicalConnector when CIM_PhysicalPackage is the container instance.

6.13.2.1.1 Command Form

```
show <CIM_PhysicalConnector multiple instances>
```

6.13.2.1.2 CIM Requirements

See CIM_PhysicalConnector in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.13.2.1.3 Behavior Requirements

6.13.2.1.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalPackage and is associated to the targeted instances of CIM_PhysicalConnector through the CIM_Container association.

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

6.13.2.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_PhysicalConnector”, “CIM_Container”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.13.2.2 Show Command Form for Multiple Instances Target – CIM_ConcreteCollection Container Instance

This command form is used to show many instances of CIM_PhysicalConnector when CIM_ConcreteCollection is the container instance.

6.13.2.2.1 Command Form

```
show <CIM_PhysicalConnector multiple instances>
```

6.13.2.2.2 CIM Requirements

See CIM_PhysicalConnector in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.13.2.2.3 Behavior Requirements

6.13.2.2.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_ConcreteCollection and is associated to the targeted instances of CIM_PhysicalConnector through the CIM_MemberOfCollection association.

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

6.13.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_PhysicalConnector”, “CIM_MemberOfCollection”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.13.2.3 Show Command Form for a Single Instance Target

This command form is used to show a single instance of CIM_PhysicalConnector.

6.13.2.3.1 Command Form

```
show <CIM_PhysicalConnector single instance>
```

6.13.2.3.2 CIM Requirements

See CIM_PhysicalConnector in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.13.2.3.3 Behavior Requirements

6.13.2.3.3.1 Preconditions

In this section `$instance` represents the targeted instance of CIM_PhysicalConnector.

```
$instance=<CIM_PhysicalConnector single instance>;
```

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

6.13.2.3.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstance ( $instance, #propertylist[] );
&smEnd;
```

6.14 CIM_PhysicalFrame

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

Table 14 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 14 is for informational purposes only; in case of a conflict between Table 14 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 14.

Table 14 – Command Verb Requirements for CIM_PhysicalFrame

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 6.14.2.
Start	Not supported	
Stop	Not supported	

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

6.14.1 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_PhysicalFrame are unordered; therefore, no algorithm is defined.

6.14.2 Show

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

6.14.2.1 Show Command Form for Multiple Instances Target – CIM_PhysicalPackage Container Instance

This command form is used to show many instances of CIM_PhysicalFrame when CIM_PhysicalPackage is the container instance.

6.14.2.1.1 Command Form

```
show <CIM_PhysicalFrame multiple instances>
```

6.14.2.1.2 CIM Requirements

See CIM_PhysicalFrame in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.14.2.1.3 Behavior Requirements

6.14.2.1.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalPackage and is associated to the targeted instances of CIM_PhysicalFrame through the CIM_Container association.

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

6.14.2.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_PhysicalFrame”, “CIM_Container”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.14.2.2 Show Command Form for Multiple Instances Target – CIM_PhysicalConnector Container Instance

This command form is used to show many instances of CIM_PhysicalFrame when CIM_PhysicalConnector is the container instance.

6.14.2.2.1 Command Form

```
show <CIM_PhysicalFrame multiple instances>
```

6.14.2.2.2 CIM Requirements

See CIM_PhysicalFrame in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties.

6.14.2.2.3 Behavior Requirements

6.14.2.2.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalConnector and is associated to the targeted instances of CIM_PhysicalFrame through the CIM_PackageInConnector association.

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

6.14.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_PhysicalFrame”, “CIM_PackageInConnector”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.14.2.3 Show Command Form for Multiple Instances Target – CIM_ConcreteCollection Container Instance

This command form is used to show many instances of CIM_PhysicalFrame when CIM_ConcreteCollection is the container instance.

6.14.2.3.1 Command Form

```
show <CIM_PhysicalFrame multiple instances>
```

6.14.2.3.2 CIM Requirements

See CIM_PhysicalFrame in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.14.2.3.3 Behavior Requirements

6.14.2.3.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_ConcreteCollection and is associated to the targeted instances of CIM_PhysicalFrame through the CIM_MemberOfCollection association.

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

6.14.2.3.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_PhysicalFrame”, “CIM_MemberOfCollection”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.14.2.4 Show Command Form for a Single Instance Target

This command form is used to show a single instance of CIM_PhysicalFrame.

6.14.2.4.1 Command Form

```
show <CIM_PhysicalFrame single instance>
```

6.14.2.4.2 CIM Requirements

See CIM_PhysicalFrame in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.14.2.4.3 Behavior Requirements

6.14.2.4.3.1 Preconditions

In this section `$instance` represents the targeted instance of CIM_PhysicalFrame.

```
$instance=<CIM_PhysicalFrame single instance>;
```

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

6.14.2.4.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstance ( $instance, #propertylist[] );
&smEnd;
```

6.15 CIM_PhysicalMemory

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

Table 15 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 15 is for informational purposes only; in case of a conflict between Table 15 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 15.

Table 15 – Command Verb Requirements for CIM_PhysicalMemory

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 6.15.2.
Start	Not supported	
Stop	Not supported	

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

6.15.1 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_PhysicalMemory are unordered; therefore, no algorithm is defined.

6.15.2 Show

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

6.15.2.1 Show Command Form for Multiple Instances Target – CIM_PhysicalPackage Container Instance

This command form is used to show many instances of CIM_PhysicalMemory when CIM_PhysicalPackage is the container instance.

6.15.2.1.1 Command Form

```
show <CIM_PhysicalMemory multiple instances>
```

6.15.2.1.2 CIM Requirements

See CIM_PhysicalMemory in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties.

6.15.2.1.3 Behavior Requirements

6.15.2.1.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalPackage and is associated to the targeted instances of CIM_PhysicalMemory through the CIM_Container association.

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

6.15.2.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( "CIM_PhysicalMemory", "CIM_Container",
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.15.2.2 Show Command Form for Multiple Instances Target – CIM_ConcreteCollection Container Instance

This command form is used to show many instances of CIM_PhysicalMemory when CIM_ConcreteCollection is the container instance.

6.15.2.2.1 Command Form

```
show <CIM_PhysicalMemory multiple instances>
```

6.15.2.2.2 CIM Requirements

See CIM_PhysicalMemory in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties.

6.15.2.2.3 Behavior Requirements

6.15.2.2.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_ConcreteCollection and is associated to the targeted instances of CIM_PhysicalMemory through the CIM_MemberOfCollection association.

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

6.15.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_PhysicalMemory”, “CIM_MemberOfCollection”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.15.2.3 Show Command Form for a Single Instance Target

This command form is used to show a single instance of CIM_PhysicalMemory.

6.15.2.3.1 Command Form

```
show <CIM_PhysicalMemory single instance>
```

6.15.2.3.2 CIM Requirements

See CIM_PhysicalMemory in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.15.2.3.3 Behavior Requirements

6.15.2.3.3.1 Preconditions

In this section `$instance` represents the targeted instance of CIM_PhysicalMemory.

```
$instance=<CIM_PhysicalMemory single instance>;
```

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

6.15.2.3.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstance ( $instance, #propertylist[] );
&smEnd;
```

6.16 CIM_PhysicalPackage

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

Table 16 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 16 is for informational purposes only; in case of a conflict between Table 16 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 16.

Table 16 – Command Verb Requirements for CIM_PhysicalPackage

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 6.16.2.
Start	Not supported	
Stop	Not supported	

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

6.16.1 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_PhysicalPackage are unordered; therefore, no algorithm is defined.

6.16.2 Show

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

6.16.2.1 Show Command Form for Multiple Instances Target – CIM_PhysicalPackage Container Instance

This command form is used to show many instances of CIM_PhysicalPackage when CIM_PhysicalPackage is the container instance.

6.16.2.1.1 Command Form

```
show <CIM_PhysicalPackage multiple instances>
```

6.16.2.1.2 CIM Requirements

See CIM_PhysicalPackage in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.16.2.1.3 Behavior Requirements

6.16.2.1.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalPackage and is associated to the targeted instances of CIM_PhysicalPackage through the CIM_Container association.

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

6.16.2.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_PhysicalPackage”, “CIM_Container”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.16.2.2 Show Command Form for Multiple Instances Target – CIM_PhysicalConnector Container Instance

This command form is used to show many instances of CIM_PhysicalPackage when CIM_PhysicalConnector is the container instance.

6.16.2.2.1 Command Form

```
show <CIM_PhysicalPackage multiple instances>
```

6.16.2.2.2 CIM Requirements

See CIM_PhysicalPackage in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.16.2.2.3 Behavior Requirements

6.16.2.2.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalConnector and is associated to the targeted instances of CIM_PhysicalPackage through the CIM_PackageInConnector association.

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

6.16.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_PhysicalPackage”, “CIM_PackageInConnector”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.16.2.3 Show Command Form for Multiple Instances Target – CIM_ConcreteCollection Container Instance

This command form is used to show many instances of CIM_PhysicalPackage when CIM_ConcreteCollection is the container instance.

6.16.2.3.1 Command Form

```
show <CIM_PhysicalPackage multiple instances>
```

6.16.2.3.2 CIM Requirements

See CIM_PhysicalPackage in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.16.2.3.3 Behavior Requirements

6.16.2.3.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_ConcreteCollection and is associated to the targeted instances of CIM_PhysicalPackage through the CIM_MemberOfCollection association.

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

6.16.2.3.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_PhysicalPackage”, “CIM_MemberOfCollection”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.16.2.4 Show Command Form for a Single Instance Target

This command form is used to show a single instance of CIM_PhysicalPackage.

6.16.2.4.1 Command Form

```
show <CIM_PhysicalPackage single instance>
```

6.16.2.4.2 CIM Requirements

See CIM_PhysicalPackage in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.16.2.4.3 Behavior Requirements

6.16.2.4.3.1 Preconditions

In this section `$instance` represents the targeted instance of CIM_PhysicalPackage.

```
$instance=<CIM_PhysicalPackage single instance>;
```

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

6.16.2.4.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstance ( $instance, #propertylist[] );
&smEnd;
```

6.17 CIM_Rack

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

Table 17 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 17 is for informational purposes only; in case of a conflict between Table 17 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 17.

Table 17 – Command Verb Requirements for CIM_Rack

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 6.17.2.
Start	Not supported	
Stop	Not supported	

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

6.17.1 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_Rack are unordered; therefore, no algorithm is defined.

6.17.2 Show

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

6.17.2.1 Show Command Form for Multiple Instances Target – CIM_PhysicalPackage Container Instance

This command form is used to show many instances of CIM_Rack when CIM_PhysicalPackage is the container instance.

6.17.2.1.1 Command Form

```
show <CIM_Rack multiple instances>
```

6.17.2.1.2 CIM Requirements

See CIM_Rack in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.17.2.1.3 Behavior Requirements

6.17.2.1.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalPackage and is associated to the targeted instances of CIM_Rack through the CIM_Container association.

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

6.17.2.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_Rack”, “CIM_Container”, $containerInstance.getInstancePath(),
   #propertylist[] );
&smEnd;
```

6.17.2.2 Show Command Form for Multiple Instances Target – CIM_PhysicalConnector Container Instance

This command form is used to show many instances of CIM_Rack when CIM_PhysicalConnector is the container instance.

6.17.2.2.1 Command Form

```
show <CIM_Rack multiple instances>
```

6.17.2.2.2 CIM Requirements

See CIM_Rack in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.17.2.2.3 Behavior Requirements

6.17.2.2.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalConnector and is associated to the targeted instances of CIM_Rack through the CIM_PackageInConnector association.

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

6.17.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_Rack”, “CIM_PackageInConnector”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.17.2.3 Show Command Form for Multiple Instances Target – CIM_ConcreteCollection Container Instance

This command form is used to show many instances of CIM_Rack when CIM_ConcreteCollection is the container instance.

6.17.2.3.1 Command Form

```
show <CIM_Rack multiple instances>
```

6.17.2.3.2 CIM Requirements

See CIM_Rack in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties.

6.17.2.3.3 Behavior Requirements

6.17.2.3.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_ConcreteCollection and is associated to the targeted instances of CIM_Rack through the CIM_MemberOfCollection association.

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

6.17.2.3.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( "CIM_Rack", "CIM_MemberOfCollection",
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.17.2.4 Show Command Form for a Single Instance Target

This command form is used to show a single instance of CIM_Rack.

6.17.2.4.1 Command Form

```
show <CIM_Rack single instance>
```

6.17.2.4.2 CIM Requirements

See CIM_Rack in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties.

6.17.2.4.3 Behavior Requirements

6.17.2.4.3.1 Preconditions

In this section `$instance` represents the targeted instance of CIM_Rack.

```
$instance=<CIM_Rack single instance>;
```

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

6.17.2.4.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstance ( $instance, #propertylist[] );
&smEnd;
```

6.18 CIM_Realizes

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

Table 18 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 18 is for informational purposes only; in case of a conflict between Table 18 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 18.

Table 18 – Command Verb Requirements for CIM_Realizes

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 6.18.2.
Start	Not supported	
Stop	Not supported	

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

6.18.1 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_Realizes are unordered; therefore, no algorithm is defined.

6.18.2 Show

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

6.18.2.1 Show Command Form for Multiple Instances Target – Subclass CIM_LogicalDevice Reference

This command form is used to show many instances of CIM_Realizes. This command form corresponds to a `show` command issued against instances of CIM_Realizes where only one reference is specified and the reference is the instance of a subclass of CIM_LogicalDevice.

6.18.2.1.1 Command Form

```
show <CIM_Realizes multiple instances>
```

6.18.2.1.2 CIM Requirements

See CIM_Realizes in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.18.2.1.3 Behavior Requirements

6.18.2.1.3.1 Preconditions

In this section `$instance` represents the instance of a subclass of CIM_LogicalDevice which is referenced by CIM_Realizes.

6.18.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( "CIM_Realizes", $instance.getInstancePath() );
&smEnd;
```

6.18.2.2 Show Command Form for Multiple Instances Target – Subclass CIM_PhysicalElement Reference

This command form is used to show many instances of CIM_Realizes. This command form corresponds to a `show` command issued against instances of CIM_Realizes where only one reference is specified and the reference is the instance of a subclass of CIM_PhysicalElement.

6.18.2.2.1 Command Form

```
show <CIM_Realizes multiple instances>
```

6.18.2.2.2 CIM Requirements

See CIM_Realizes in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.18.2.2.3 Behavior Requirements

6.18.2.2.3.1 Preconditions

In this section `$instance` represents the instance of a subclass of CIM_PhysicalElement which is referenced by CIM_Realizes.

6.18.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( "CIM_Realizes", $instance.getInstancePath() );
&smEnd;
```

6.18.2.3 Show Command Form for 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_Realizes where both references are specified and therefore the desired instance is unambiguously identified.

6.18.2.3.1 Command Form

```
show <CIM_Realizes single instance>
```

6.18.2.3.2 CIM Requirements

See CIM_Realizes in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.18.2.3.3 Behavior Requirements

6.18.2.3.3.1 Preconditions

In this section `$instanceA` represents the referenced instance of a subclass of CIM_LogicalDevice through CIM_Realizes association. `$instanceB` represents the instance of a subclass of CIM_PhysicalElement which is referenced by CIM_Realizes.

6.18.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_Realizes”, $instanceA.getInstancePath(),
   $instanceB.getInstancePath() );
&smEnd;
```

6.19 CIM_Slot

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

Table 19 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 19 is for informational purposes only; in case of a conflict between Table 19 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 19.

Table 19 – Command Verb Requirements for CIM_Slot

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 6.19.2.
Start	Not supported	
Stop	Not supported	

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

6.19.1 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_Slot are unordered; therefore, no algorithm is defined.

6.19.2 Show

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

6.19.2.1 Show Command Form for Multiple Instances Target – CIM_PhysicalPackage Container Instance

This command form is used to show many instances of CIM_Slot when CIM_PhysicalPackage is the container instance.

6.19.2.1.1 Command Form

```
show <CIM_Slot multiple instances>
```

6.19.2.1.2 CIM Requirements

See CIM_Slot in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.19.2.1.3 Behavior Requirements

6.19.2.1.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_PhysicalPackage and is associated to the targeted instances of CIM_Slot through the CIM_Container association.

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

6.19.2.1.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_Slot”, “CIM_Container”, $containerInstance.getInstancePath(),
   #propertylist[] );
&smEnd;
```

6.19.2.2 Show Command Form for Multiple Instances Target – CIM_ConcreteCollection Container Instance

This command form is used to show many instances of CIM_Slot when CIM_ConcreteCollection is the container instance.

6.19.2.2.1 Command Form

```
show <CIM_Slot multiple instances>
```

6.19.2.2.2 CIM Requirements

See CIM_Slot in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.19.2.2.3 Behavior Requirements

6.19.2.2.3.1 Preconditions

In this section `$containerInstance` represents the instance of CIM_ConcreteCollection and is associated to the targeted instances of CIM_Slot through the CIM_MemberOfCollection association.

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

6.19.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstances ( “CIM_Slot”, “CIM_MemberOfCollection”,
   $containerInstance.getInstancePath(), #propertylist[] );
&smEnd;
```

6.19.2.3 Show Command Form for a Single Instance Target

This command form is used to show a single instance of CIM_Slot.

6.19.2.3.1 Command Form

```
show <CIM_Slot single instance>
```

6.19.2.3.1.1 CIM Requirements

See CIM_Slot in the “CIM Elements” section of the *Physical Asset Profile* for the list of mandatory properties.

6.19.2.3.2 Behavior Requirements

6.19.2.3.2.1 Preconditions

In this section `$instance` represents the targeted instance of CIM_Slot.

```
$instance=<CIM_Slot single instance>;
```

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

6.19.2.3.2.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all) {
   #propertylist[] = <array of mandatory non-key property names (see CIM
      Requirements)>;
}
&smShowInstance ( $instance, #propertylist[] );
&smEnd;
```

6.20 CIM_SystemPackaging

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

Table 20 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 20 is for informational purposes only; in case of a conflict between Table 20 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 20.

Table 20 – Command Verb Requirements for CIM_SystemPackaging

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 6.20.2.
Start	Not supported	
Stop	Not supported	

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

6.20.1 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_SystemPackaging are unordered; therefore, no algorithm is defined.

6.20.2 Show

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

6.20.2.1 Show Command Form for Multiple Instances Target – CIM_System

This command form is used to show many instances of CIM_SystemPackaging. This command form corresponds to a `show` command issued against CIM_SystemPackaging where only one reference is specified and the reference is to an instance of CIM_System.

6.20.2.1.1 Command Form

```
show <CIM_SystemPackaging multiple instances>
```

6.20.2.1.2 CIM Requirements

See CIM_SystemPackaging in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.20.2.1.3 Behavior Requirements

6.20.2.1.3.1 Preconditions

In this section `$instance` represents the instance of CIM_System which is referenced by CIM_SystemPackaging.

6.20.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( "CIM_SystemPackaging", $instance.getInstancePath() );
&smEnd;
```

6.20.2.2 Show Command Form for Multiple Instances – CIM_PhysicalPackage Reference

This command form is used to show multiple instances of CIM_SystemPackaging. This command form corresponds to a `show` command issued against multiple instances of CIM_SystemPackaging where only one reference is specified and the reference is to an instance of CIM_PhysicalPackage or a subclass of CIM_PhysicalPackage.

6.20.2.2.1 Command Form

```
show <CIM_SystemPackaging multiple instances>
```

6.20.2.2.2 CIM Requirements

See CIM_SystemPackaging in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.20.2.2.3 Behavior Requirements

6.20.2.2.3.1 Preconditions

In this section `$instance` represents the instance of CIM_PhysicalPackage or a subclass of CIM_PhysicalPackage which is referenced by CIM_SystemPackaging.

6.20.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( "CIM_SystemPackaging", $instance.getInstancePath() );
&smEnd;
```

6.20.2.3 Show Command Form for 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_SystemPackaging where both references are specified and therefore the desired instance is unambiguously identified.

6.20.2.3.1 Command Form

```
show <CIM_SystemPackaging single instance>
```

6.20.2.3.2 CIM Requirements

See CIM_SystemPackaging in the "CIM Elements" section of the *Physical Asset Profile* for the list of mandatory properties and CIM classes that can be referenced.

6.20.2.3.3 Behavior Requirements

6.20.2.3.3.1 Preconditions

In this section `$instanceA` represents the referenced instance of CIM_System through CIM_SystemPackaging association. `$instanceB` represents the instance of CIM_PhysicalPackage or a subclass of CIM_PhysicalPackage which is referenced by CIM_SystemPackaging.

6.20.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_SystemPackaging”, $instanceA.getInstancePath(),
   $instanceB.getInstancePath() );
&smEnd;
```

ANNEX A
(informative)

Change Log

Version	Date	Author	Description
1.0.0	2009-07-14		DMTF Standard Release