**Document Number: DSP0802**

**Date: 2009-06-04**

**Version: 1.0.0**

# SMASH Collections Profile SM CLP Command Mapping Specification

**Document Type: Specification**

**Document Status: DMTF Standard**

**Document Language: E**

# CONTENTS

## Tables

# Foreword

The *SMASH Collections Profile SM CLP Command Mapping Specification* (DSP0802) was prepared by the Server Management Working Group.

## Conventions

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

## Acknowledgements

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

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

# Introduction

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

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

# SMASH Collections Profile SM CLP Command Mapping Specification

# 1 Scope

This specification contains the requirements for an implementation of the SM CLP to provide access to, and implement the behaviors of, the *SMASH Collections 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 DSP1006, *SMASH Collections Profile 1.0*, http://www.dmtf.org/standards/published_documents/DSP1006_1.0.pdf

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

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

Other References

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

# 3 Terms and Definitions

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

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

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

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

**3.4**

**mandatory**

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

**3.5**

**may**

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

**3.6**

**need not**

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

**3.7**

**optional**

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

**3.8**

**shall**

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

**3.9**

**shall not**

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

**3.10**

**should**

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

**3.11**

**should not**

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

# 4 Symbols and Abbreviated Terms

The following symbols and abbreviations are used in this document.

**4.1**

**CIM**

Common Information Model

**4.2**

**CLP**

Command Line Protocol

**4.3**

**DMTF**

Distributed Management Task Force

**4.4**
**IETF**
Internet Engineering Task Force

**4.5**
**SM**
Server Management

**4.6**
**SMI-S**
Storage Management Initiative Specification

**4.7**
**SNIA**
Storage Networking Industry Association

# 5 Recipes

The following is a list of the common recipes used by the mappings in this specification. For a definition of each recipe, see the *SM CLP-to-CIM Common Mapping Specification 1.0* (DSP0216).

- smShowInstance()
- smShowInstances()
- smSetInstance()
- smShowAssociationInstances()
- smShowAssociationInstance()

This mapping does not define any recipes for local reuse.

# 6 Mappings

The following sections detail the mapping of CLP verbs to CIM Operations for each CIM class defined in the *SMASH Collections Profile* (DSP1006). Requirements specified here related to support for a CLP verb for a particular class are solely within the context of this profile.

## 6.1 CIM_ConcreteCollection

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

Table 1 lists each SM CLP verb, the required level of support for the verb in conjunction with 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_ConcreteCollection**

| Command Verb | Requirement | Comments |
|---|---|---|
| create | Not supported | None |
| delete | Not supported | None |
| dump | Not supported | None |
| load | Not supported | None |
| reset | Not supported | None |
| set | Not supported | None |
| show | Shall | See 6.1.2 |
| start | Not supported | None |
| stop | Not supported | None |

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

### 6.1.1 Ordering of Results

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

- Results for CIM_ConcreteCollection 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_ConcreteCollection. Implementations shall support the use of the `show` verb with CIM_ConcreteCollection.

#### 6.1.2.1 Show a Single Instance of CIM_ConcreteCollection

##### 6.1.2.1.1 Command Form

```
show <CIM_ConcreteCollection single instance>
```

##### 6.1.2.1.2 CIM Requirements

See the "CIM Elements" section of the *SMASH Collections Profile*.

##### 6.1.2.1.3 Behavior Requirements

###### 6.1.2.1.3.1 Preconditions

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

###### 6.1.2.1.3.2 Pseudo Code

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

#### 6.1.2.2 Show Multiple Instances of CIM_ConcreteCollection

##### 6.1.2.2.1 Command Form

```
show <CIM_ConcreteCollection multiple instances>
```

##### 6.1.2.2.2 CIM Requirements

See the "CIM Elements" section of the *SMASH Collections Profile*.

##### 6.1.2.2.3 Behavior Requirements

###### 6.1.2.2.3.1 Preconditions

`$containerInstance` contains the instance of CIM_ComputerSystem for which we are displaying related CIM_ConcreteCollection instances.

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

###### 6.1.2.2.3.2 Pseudo Code

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

### 6.2 CIM_MemberOfCollection

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

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

**Table 2 – Command Verb Requirements for CIM_MemberOfCollection**

| Command Verb | Requirement | Comments |
|---|---|---|
| create | Not supported | None |
| delete | Not supported | None |
| dump | Not supported | None |
| load | Not supported | None |
| reset | Not supported | None |
| set | Not supported | None |
| show | Shall | See 6.2.2 |
| start | Not supported | None |
| stop | Not supported | None |

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

### 6.2.1 Ordering of Results

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

- Results for CIM_MemberOfCollection 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_MemberOfCollection. Implementations shall support the use of the `show` verb with CIM_MemberOfCollection.

#### 6.2.2.1 Show a Single Instance – Both References

This command form is for the `show` command applied to CIM_MemberOfCollection where both references are specified. Therefore, exactly one instance is shown.

##### 6.2.2.1.1 Command Form

```
show <CIM_MemberOfCollection single instance>
```

##### 6.2.2.1.2 CIM Requirements

See the "CIM Elements" section of the *SMASH Collections Profile*.

##### 6.2.2.1.3 Behavior Requirements

###### 6.2.2.1.3.1 Preconditions

`$instanceA` contains one of the instances of CIM_ConcreteCollection referenced by CIM_MemberOfCollection.

`$instanceB` contains one of the instances of CIM_ManagedElement referenced by CIM_MemberOfCollection.

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

###### 6.2.2.1.3.2 Psuedo Code

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

#### 6.2.2.2 Show Multiple Instances – CIM_ConcreteCollection Reference

This command form is for the `show` command applied to an instance of CIM_MemberOfCollection where only the reference to an instance of CIM_ConcreteCollection is specified. Zero or more instances of CIM_MemberOfCollection can reference a single instance of CIM_ConcreteCollection.

###### 6.2.2.2.1 Command Form

```
show <CIM_MemberOfCollection multiple instances>
```

###### 6.2.2.2.2 CIM Requirements

See the "CIM Elements" section of the *SMASH Collections Profile*.

###### 6.2.2.2.3 Behavior Requirements

###### 6.2.2.2.3.1 Preconditions

`$instance` contains the instance of CIM_ConcreteCollection that is referenced by CIM_MemberOfCollection

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

###### 6.2.2.2.3.2 Psuedo Code

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

#### 6.2.2.3 Show a Single Instance – CIM_ManagedElement Reference

This command form is for the `show` command applied to CIM_MemberOfCollection where only the reference to an instance of CIM_ManagedElement is specified. An instance of CIM_ManagedElement can be referenced by at most one instance of CIM_MemberOfCollection.

##### 6.2.2.3.1 Command Form

```
show <CIM_MemberOfCollection single instances>
```

##### 6.2.2.3.2 CIM Requirements

See the "CIM Elements" section of the *SMASH Collections Profile*.

##### 6.2.2.3.3 Behavior Requirements

###### 6.2.2.3.3.1 Preconditions

`$instance` contains the instance of CIM_ManagedElement that is referenced by CIM_MemberOfCollection

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

###### 6.2.2.3.3.2 Psuedo Code

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

## 6.3 CIM_OwningCollectionElement

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

Table 3 lists each SM CLP verb, the required level of support for the verb in conjunction with 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_OwningCollectionElement**

| Command Verb | Requirement | Comments |
|---|---|---|
| create | Not supported | None |
| delete | Not supported | None |
| dump | Not supported | None |
| load | Not supported | None |
| reset | Not supported | None |
| set | Not supported | None |
| show | Shall | See 6.3.2. |
| start | Not supported | None |
| stop | Not supported | None |

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

### 6.3.1 Ordering of Results

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

- Results for CIM_OwningCollectionElement 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_OwningCollectionElement. Implementations shall support the use of the `show` verb with CIM_OwningCollectionElement.

#### 6.3.2.1 Show a Single Instance – Both References

This command form is for the `show` command applied to CIM_OwningCollectionElement where both references are specified. Therefore, a single instance will be returned.

##### 6.3.2.1.1 Command Form

```
show <CIM_OwningCollectionElement single instance>
```

##### 6.3.2.1.2 CIM Requirements

See the “CIM Elements” section of the *SMASH Collections Profile*.

###### 6.3.2.1.3 Behavior Requirements

###### 6.3.2.1.3.1 Preconditions

`$instanceA` contains one of the instances of <CIM_ConcreteCollection> that is referenced by CIM_OwningCollectionElement

`$instanceB` contains one of the instances of <CIM_ComputerSystem> that referenced by CIM_OwningCollectionElement.

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

### 6.3.2.2 Show Multiple Instances – CIM_ComputerSystem Reference

This command form is for the `show` command applied to CIM_OwningCollectionElement where only the reference to an instance of CIM_ComputerSystem is specified. An instance of CIM_ComputerSystem can be referenced by multiple instances of CIM_OwningCollectionElement.

#### 6.3.2.2.1 Command Form

```
show <CIM_OwningCollectionElement multiple instances>
```

#### 6.3.2.2.2 CIM Requirements

See the “CIM Elements” section of the *SMASH Collections Profile*.

#### 6.3.2.2.3 Behavior Requirements

###### 6.3.2.2.3.1 Preconditions

`$instance` contains the instance of <CIM_ComputerSystem> that is referenced by CIM_OwningCollectionElement

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

#### 6.3.2.3 Show a Single Instance – CIM_ConcreteCollection Reference

This command form is for the `show` command applied to CIM_OwningCollectionElement where only the reference to an instance of CIM_ConcreteCollection is specified. An instance of CIM_ConcreteCollection is referenced by exactly one instance of CIM_OwningCollectionElement. Therefore, a single instance is returned.

##### 6.3.2.3.1 Command Form

```
show <CIM_OwningCollectionElement single instances>
```

##### 6.3.2.3.2 CIM Requirements

See the "CIM Elements" section of the *SMASH Collections Profile*.

##### 6.3.2.3.3 Behavior Requirements

###### 6.3.2.3.3.1 Preconditions

`$instance` contains the instance of <CIM_ConcreteCollection> that is referenced by CIM_OwningCollectionElement

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

## ANNEX A
(informative)

## Change Log

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