Document Number: DSP0815

Date: 2009-06-04

Version: 1.0.0

Ethernet Port Profile SM CLP Command Mapping Specification

Document Type: Specification

Document Status: DMTF Standard

Document Language: E

CONTENTS

Tables

50

Foreword

The *Ethernet Port Profile SM CLP Command Mapping Specification* (DSP0815) 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 *Ethernet Port 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 *Ethernet Port Profile* using CIM operations.

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

Ethernet Port 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 *Ethernet Port 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 DSP1014, *Ethernet Port Profile 1..0,*
http://www.dmtf.org/standards/published_documents/DSP1014_1.0.pdf

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

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

2.2 Other References

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

3 Terms and Definitions

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

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

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

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

3.4
mandatory
indicates requirements to be followed strictly in order to conform to the document and from which no deviation is permitted

3.5
may
indicates a course of action permissible within the limits of the document

3.6
need not
indicates a course of action permissible within the limits of the document

3.7
optional
indicates a course of action permissible within the limits of the document

3.8
shall
indicates requirements to be followed strictly in order to conform to the document and from which no deviation is permitted

3.9
shall not
indicates requirements to be followed strictly in order to conform to the document and from which no deviation is permitted

3.10
should
indicates that among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others, or that a certain course of action is preferred but not necessarily required

3.11
should not
indicates that a certain possibility or course of action is deprecated but not prohibited

4 Symbols and Abbreviated Terms

The following symbols and abbreviations are used in this document.

4.1
CIM
Common Information Model

4.2
CLP
Command Line Protocol

4.3
DMTF
Distributed Management Task Force

4.4
IETF
Internet Engineering Task Force

4.5
SM
Server Management

4.6
SMI-S
Storage Management Initiative Specification

4.7
SNIA
Storage Networking Industry Association

4.8
UFsT
User Friendly selection Tag

5 Recipes

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

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

This mapping does not define any recipes for local reuse.

6 Mappings

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

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

Table 1 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 1 is for informational purposes only; in case of a conflict between Table 1 and

requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 1.

Table 1 – Command Verb Requirements for CIM_EthernetPort

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	May	See 6.1.2.
set	May	See 6.1.3.
show	Shall	See 6.1.5.
start	May	See 6.1.6.
stop	May	See 6.1.7.

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

6.1.1 Ordering of Results

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

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

6.1.2 Reset

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

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

6.1.2.1 Reset a Single Instance

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

6.1.2.1.1 Command Form

```
reset <CIM_EthernetPort single object>
```

6.1.2.1.2 CIM Requirements

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

6.1.2.1.3 Behavior Requirements

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

6.1.3 Set

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

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

6.1.3.1 General Usage of Set for a Single Property

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

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

6.1.4 Command Form

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

6.1.4.1.1 CIM Requirements

See CIM_EthernetPort in the “CIM Elements” section of the *Ethernet Port Profile* for the list of modifiable properties.

6.1.4.1.2 Behavior Requirements

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

6.1.4.2 General Usage of Set for Multiple Properties

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

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

6.1.4.2.1 Command Form

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

6.1.4.2.2 CIM Requirements

See CIM_EthernetPort in the “CIM Elements” section of the *Ethernet Port Profile* for the list of mandatory properties.

6.1.4.2.3 Behavior Requirements

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

6.1.5 Show

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

The `show` verb is used to display information about the Ethernet port.

6.1.5.1 Show a Single Instance

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

6.1.5.1.1 Command Form

```
show <CIM_EthernetPort single object>
```

6.1.5.1.2 Behavior Requirements

```
$instance=<CIM_EthernetPort single object>
#propertylist[] = NULL;
if (false == #all)
   {
   #propertylist[] = {“LinkTechnology”, “PermanentAddress”, “DeviceID”, “ElementName”,
      “EthernetAddresses”, “Capabilities”, “EnabledCapabilities”};
   }
&smShowInstance ( $instance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.1.5.1.2.1 Preconditions

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

6.1.5.1.2.2 Pseudo Code

```
$instance=<CIM_EthernetPort single object>
#propertylist[] = NULL;
if (false == #all)
   {
   #propertylist[] = {“LinkTechnology”, “PermanentAddress”, “DeviceID”, “ElementName”,
      “EthernetAddresses”, “Capabilities”, “EnabledCapabilities”};
   }
&smShowInstance ( $instance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.1.5.2 Show Multiple Instances

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

6.1.5.2.1 Command Form

```
show <CIM_EthernetPort multiple objects>
```

6.1.5.2.2 Behavior Requirements

```
#propertylist[] = NULL;
if (false == #all)
   {
   #propertylist[] = {“LinkTechnology”, “PermanentAddress”, “DeviceID”, “ElementName”,
      “EthernetAddresses”, “Capabilities”, “EnabledCapabilities”};
   }
&smShowInstances ( “CIM_EthernetPort”, “CIM_SystemDevice”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.1.5.2.2.1 Preconditions

$containerInstance contains the instance of CIM_ComputerSystem for which we are displaying scoped Ethernet ports (CIM_EthernetPort instances). The *Ethernet Port Profile* requires that the CIM_EthernetPort instance be associated with its scoping system via an instance of the CIM_SystemDevice association.

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

6.1.5.2.2.2 Pseudo Code

```
#propertylist[] = NULL;
if (false == #all)
   {
   #propertylist[] = {“LinkTechnology”, “PermanentAddress”, “DeviceID”, “ElementName”,
      “EthernetAddresses”, “Capabilities”, “EnabledCapabilities”};
   }
&smShowInstances ( “CIM_EthernetPort”, “CIM_SystemDevice”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.1.6 Start

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

The `start` verb is used to enable a Ethernet port.

6.1.6.1 Start a Single Instance

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

6.1.6.1.1 Command Form

```
start <CIM_EthernetPort single object>
```

6.1.6.1.2 CIM Requirements

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

6.1.6.1.3 Behavior Requirements

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

6.1.7 Stop

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

The `stop` verb is used to disable a Ethernet port.

6.1.7.1 Stop a Single Instance

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

6.1.7.1.1 Command Form

```
stop <CIM_EthernetPort single object>
```

6.1.7.1.2 CIM Requirements

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

6.1.7.1.3 Behavior Requirements

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

ANNEX A
(informative)

Change Log

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