



1
2
3
4

Document Number: DSP0815

Date: 2009-06-04

Version: 1.0.0

5 **Ethernet Port Profile SM CLP Command Mapping**
6 **Specification**

7 **Document Type: Specification**
8 **Document Status: DMTF Standard**
9 **Document Language: E**

10

11 Copyright notice

12 Copyright © 2006, 2009 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

13 DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems
14 management and interoperability. Members and non-members may reproduce DMTF specifications and
15 documents, provided that correct attribution is given. As DMTF specifications may be revised from time to
16 time, the particular version and release date should always be noted.

17 Implementation of certain elements of this standard or proposed standard may be subject to third party
18 patent rights, including provisional patent rights (herein "patent rights"). DMTF makes no representations
19 to users of the standard as to the existence of such rights, and is not responsible to recognize, disclose,
20 or identify any or all such third party patent right, owners or claimants, nor for any incomplete or
21 inaccurate identification or disclosure of such rights, owners or claimants. DMTF shall have no liability to
22 any party, in any manner or circumstance, under any legal theory whatsoever, for failure to recognize,
23 disclose, or identify any such third party patent rights, or for such party's reliance on the standard or
24 incorporation thereof in its product, protocols or testing procedures. DMTF shall have no liability to any
25 party implementing such standard, whether such implementation is foreseeable or not, nor to any patent
26 owner or claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is
27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party
28 implementing the standard from any and all claims of infringement by a patent owner for such
29 implementations.

30 For information about patents held by third-parties which have notified the DMTF that, in their opinion,
31 such patent may relate to or impact implementations of DMTF standards, visit
32 <http://www.dmtf.org/about/policies/disclosures.php>.

33

CONTENTS

34	Foreword	5
35	Introduction	6
36	1 Scope	7
37	2 Normative References.....	7
38	2.1 Approved References	7
39	2.2 Other References.....	7
40	3 Terms and Definitions.....	7
41	4 Symbols and Abbreviated Terms.....	8
42	5 Recipes.....	9
43	6 Mappings.....	9
44	6.1 CIM_EthernetPort	9

45

46

47

48

49

51

Foreword

52 The *Ethernet Port Profile SM CLP Command Mapping Specification* (DSP0815) was prepared by the
53 Server Management Working Group.

54 Conventions

55 The pseudo-code conventions utilized in this document are the Recipe Conventions as defined in SNIA
56 [SMI-S 1.1.0](#), section 7.6.

57 Acknowledgements

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

- 60 • Aaron Merkin – IBM
- 61 • Jon Hass – Dell
- 62 • Khachatur Papanyan – Dell
- 63 • Jeff Hilland – HP
- 64 • Christina Shaw – HP
- 65 • Perry Vincent – Intel
- 66 • John Leung – Intel

67

68

Introduction

69 This document defines the SM CLP mapping for CIM elements described in the [Ethernet Port Profile](#). The
70 information in this specification, combined with the [SM CLP-to-CIM Common Mapping Specification 1.0](#),
71 is intended to be sufficient to implement SM CLP commands relevant to the classes, properties and
72 methods described in the [Ethernet Port Profile](#) using CIM operations.

73 The target audience for this specification is implementers of the SM CLP support for the [Ethernet Port](#)
74 [Profile](#).

75
76

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

- 106 **3.4**
107 **mandatory**
108 indicates requirements to be followed strictly in order to conform to the document and from which no
109 deviation is permitted
- 110 **3.5**
111 **may**
112 indicates a course of action permissible within the limits of the document
- 113 **3.6**
114 **need not**
115 indicates a course of action permissible within the limits of the document
- 116 **3.7**
117 **optional**
118 indicates a course of action permissible within the limits of the document
- 119 **3.8**
120 **shall**
121 indicates requirements to be followed strictly in order to conform to the document and from which no
122 deviation is permitted
- 123 **3.9**
124 **shall not**
125 indicates requirements to be followed strictly in order to conform to the document and from which no
126 deviation is permitted
- 127 **3.10**
128 **should**
129 indicates that among several possibilities, one is recommended as particularly suitable, without
130 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 131 **3.11**
132 **should not**
133 indicates that a certain possibility or course of action is deprecated but not prohibited

134 **4 Symbols and Abbreviated Terms**

135 The following symbols and abbreviations are used in this document.

- 136 **4.1**
137 **CIM**
138 Common Information Model
- 139 **4.2**
140 **CLP**
141 Command Line Protocol
- 142 **4.3**
143 **DMTF**
144 Distributed Management Task Force

145 **4.4**
146 **IETF**
147 Internet Engineering Task Force

148 **4.5**
149 **SM**
150 Server Management

151 **4.6**
152 **SMI-S**
153 Storage Management Initiative Specification

154 **4.7**
155 **SNIA**
156 Storage Networking Industry Association

157 **4.8**
158 **UFsT**
159 User Friendly selection Tag

160 **5 Recipes**

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

- 163 • smStartRSC()
- 164 • smStopRSC()
- 165 • smResetRSC()
- 166 • smShowInstance()
- 167 • smShowInstances()
- 168 • smSetInstance()
- 169 • smShowAssociationInstances()
- 170 • smShowAssociationInstance()

171 This mapping does not define any recipes for local reuse.

172 **6 Mappings**

173 The following sections detail the mapping of CLP verbs to CIM Operations for each CIM class defined in
174 the [Ethernet Port Profile](#). Requirements specified here related to support for a CLP verb for a particular
175 class are solely within the context of this profile.

176 **6.1 CIM_EthernetPort**

177 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

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

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

183 **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.

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

185 6.1.1 Ordering of Results

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

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

189 6.1.2 Reset

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

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

193 6.1.2.1 Reset a Single Instance

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

196 6.1.2.1.1 Command Form

197 `reset <CIM_EthernetPort single object>`

198 6.1.2.1.2 CIM Requirements

```

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

205 6.1.2.1.3 Behavior Requirements

```
206 $instance=<CIM_EthernetPort single object>
207 smResetRSC ( $instance.getObjectPath() );
208 &smEnd;
```

209 6.1.3 Set

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

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

213 6.1.3.1 General Usage of Set for a Single Property

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

216 The requirement for supporting modification of a property using this command form shall be equivalent to
217 the requirement for supporting modification of the property using the ModifyInstance operation as defined
218 in the [Ethernet Port Profile](#).

219 6.1.4 Command Form

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

221 6.1.4.1.1 CIM Requirements

222 See CIM_EthernetPort in the “CIM Elements” section of the [Ethernet Port Profile](#) for the list of modifiable
223 properties.

224 6.1.4.1.2 Behavior Requirements

```
225 $instance=<CIM_EthernetPort single instance>
226 #propertyName[] = {<propertyname>};
227 #propertyValues[] = {<propertyvalue>};
228 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
229 &smEnd;
```

230 6.1.4.2 General Usage of Set for Multiple Properties

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

234 The requirement for supporting modification of a property using this command form shall be equivalent to
235 the requirement for supporting modification of the property using the ModifyInstance operation as defined
236 in the [Ethernet Port Profile](#).

237 6.1.4.2.1 Command Form

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

240 6.1.4.2.2 CIM Requirements

241 See CIM_EthernetPort in the “CIM Elements” section of the [Ethernet Port Profile](#) for the list of mandatory
242 properties.

243 6.1.4.2.3 Behavior Requirements

```

244 $instance=<CIM_EthernetPort single instance>
245 #propertyName[] = {<propertyname>;}
246 for #i < n
247     {
248         #propertyName[#i] = <propertyname#i>
249         #propertyValues[#i] = <propertyvalue#i>
250     }
251 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
252 &smEnd;
```

253 6.1.5 Show

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

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

257 6.1.5.1 Show a Single Instance

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

259 6.1.5.1.1 Command Form

```

260 show <CIM_EthernetPort single object>
```

261 6.1.5.1.2 Behavior Requirements

```

262 $instance=<CIM_EthernetPort single object>
263 #propertylist[] = NULL;
264 if (false == #all)
265     {
266         #propertylist[] = {"LinkTechnology", "PermanentAddress", "DeviceID", "ElementName",
267             "EthernetAddresses", "Capabilities", "EnabledCapabilities"};
268     }
269 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
270 &smEnd;
```

271 6.1.5.1.2.1 Preconditions

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

273 6.1.5.1.2.2 Pseudo Code

```

274 $instance=<CIM_EthernetPort single object>
275 #propertylist[] = NULL;
276 if (false == #all)
277     {
278         #propertylist[] = {"LinkTechnology", "PermanentAddress", "DeviceID", "ElementName",
279             "EthernetAddresses", "Capabilities", "EnabledCapabilities"};
280     }
281 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
282 &smEnd;
```

283 6.1.5.2 Show Multiple Instances

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

286 6.1.5.2.1 Command Form

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

288 6.1.5.2.2 Behavior Requirements

```
289 #propertylist[] = NULL;
290 if (false == #all)
291 {
292     #propertylist[] = {"LinkTechnology", "PermanentAddress", "DeviceID", "ElementName",
293         "EthernetAddresses", "Capabilities", "EnabledCapabilities"};
294 }
295 &smShowInstances ( "CIM_EthernetPort", "CIM_SystemDevice",
296     $containerInstance.getObjectPath(), #propertylist[] );
297 &smEnd;
```

298 6.1.5.2.2.1 Preconditions

299 `$containerInstance` contains the instance of `CIM_ComputerSystem` for which we are displaying scoped
300 Ethernet ports (`CIM_EthernetPort` instances). The [Ethernet Port Profile](#) requires that the
301 `CIM_EthernetPort` instance be associated with its scoping system via an instance of the
302 `CIM_SystemDevice` association.

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

304 6.1.5.2.2.2 Pseudo Code

```
305 #propertylist[] = NULL;
306 if (false == #all)
307 {
308     #propertylist[] = {"LinkTechnology", "PermanentAddress", "DeviceID", "ElementName",
309         "EthernetAddresses", "Capabilities", "EnabledCapabilities"};
310 }
311 &smShowInstances ( "CIM_EthernetPort", "CIM_SystemDevice",
312     $containerInstance.getObjectPath(), #propertylist[] );
313 &smEnd;
```

314 6.1.6 Start

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

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

318 6.1.6.1 Start a Single Instance

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

320 6.1.6.1.1 Command Form

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

322 **6.1.6.1.2 CIM Requirements**

```

323 uint16 EnabledState;
324 uint16 RequestedState;
325 uint32 EnabledLogicalElement.RequestStateChange (
326     [IN] uint16 RequestedState = "<request value>",
327     [OUT] REF CIM_ConcreteJob Job,
328     [IN] datetime TimeoutPeriod );

```

329 **6.1.6.1.3 Behavior Requirements**

```

330 $instance=<CIM_EthernetPort single object>
331 smStartRSC ( $instance.getObjectPath() );
332 &smEnd;

```

333 **6.1.7 Stop**

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

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

337 **6.1.7.1 Stop a Single Instance**

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

339 **6.1.7.1.1 Command Form**

```

340 stop <CIM_EthernetPort single object>

```

341 **6.1.7.1.2 CIM Requirements**

```

342 uint16 EnabledState;
343 uint16 RequestedState;
344 uint32 EnabledLogicalElement.RequestStateChange (
345     [IN] uint16 RequestedState = "<request value>",
346     [OUT] REF CIM_ConcreteJob Job,
347     [IN] datetime TimeoutPeriod );

```

348 **6.1.7.1.3 Behavior Requirements**

```

349 $instance=<CIM_EthernetPort single object>
350 smStopRSC ( $instance.getObjectPath() );
351 &smEnd;

```

352

353
 354
 355
 356
 357

ANNEX A
 (informative)

Change Log

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

358