



1
2
3
4

Document Number: DSP0822

Date: 2009-06-04

Version: 1.0.0

5 **Power Supply Profile SM CLP Command**
6 **Mapping 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27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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

34

CONTENTS

| | | |
|----|--|----|
| 35 | Foreword | 5 |
| 36 | Introduction | 6 |
| 37 | 1 Scope | 7 |
| 38 | 2 Normative References..... | 7 |
| 39 | 2.1 Approved References | 7 |
| 40 | 2.2 Other References..... | 7 |
| 41 | 3 Terms and Definitions..... | 7 |
| 42 | 4 Symbols and Abbreviated Terms..... | 8 |
| 43 | 5 Recipes..... | 9 |
| 44 | 6 Mappings..... | 9 |
| 45 | 6.1 CIM_ElementCapabilities | 9 |
| 46 | 6.2 CIM_EnabledLogicalElementCapabilities..... | 12 |
| 47 | 6.3 CIM_IsSpare | 14 |
| 48 | 6.4 CIM_MemberOfCollection | 16 |
| 49 | 6.5 CIM_PowerSupply | 19 |
| 50 | 6.6 CIM_RedundancySet..... | 24 |
| 51 | 6.7 CIM_SystemDevice | 30 |
| 52 | 6.8 CIM_OwningCollectionElement..... | 32 |
| 53 | ANNEX A (informative) Change Log..... | 35 |
| 54 | | |

55 Tables

| | | |
|----|--|----|
| 56 | Table 1 – Command Verb Requirements for CIM_ElementCapabilities | 10 |
| 57 | Table 2 – Command Verb Requirements for CIM_EnabledLogicalElementCapabilities..... | 12 |
| 58 | Table 3 – Command Verb Requirements for CIM_IsSpare | 14 |
| 59 | Table 4 – Command Verb Requirements for CIM_MemberOfCollection | 17 |
| 60 | Table 5 – Command Verb Requirements for CIM_PowerSupply | 19 |
| 61 | Table 6 – Command Verb Requirements for CIM_RedundancySet..... | 24 |
| 62 | Table 7 – Command Verb Requirements for CIM_SystemDevice | 30 |
| 63 | Table 8 – Command Verb Requirements for CIM_OwningCollectionElement..... | 32 |
| 64 | | |

66

Foreword

67 The *Power Supply Profile SM CLP Command Mapping Specification* (DSP0822) was prepared by the
68 Server Management Working Group.

69 **Conventions**

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

72 **Acknowledgements**

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

- 75 • Khachatur Papanyan – Dell
- 76 • Jon Hass – Dell
- 77 • Jeff Hilland – HP
- 78 • Christina Shaw – HP
- 79 • Aaron Merkin – IBM
- 80 • Jeff Lynch – IBM
- 81 • Perry Vincent – Intel
- 82 • John Leung – Intel

83

84

Introduction

85 This document defines the SM CLP mapping for CIM elements described in the [Power Supply Profile](#).
86 The information in this specification, combined with *SM CLP-to-CIM Common Mapping Specification 1.0*
87 ([DSP0216](#)), is intended to be sufficient to implement SM CLP commands relevant to the classes,
88 properties and methods described in the [Power Supply Profile](#) using CIM operations.

89 The target audience for this specification is implementers of the SM CLP support for the [Power Supply](#)
90 [Profile](#).

91
92

Power Supply Profile SM CLP Command Mapping Specification

93 1 Scope

94 This specification contains the requirements for an implementation of the SM CLP to provide access to,
95 and implement the behaviors of, the [Power Supply Profile](#).

96 2 Normative References

97 The following referenced documents are indispensable for the application of this document. For dated
98 references, only the edition cited applies. For undated references, the latest edition of the referenced
99 document (including any amendments) applies.

100 2.1 Approved References

101 DMTF DSP1015, *Power Supply Profile 1.1*,
102 http://www.dmtf.org/standards/published_documents/DSP1015_1.1.pdf

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

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

107 2.2 Other References

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

110 3 Terms and Definitions

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

112 3.1

113 **can**

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

115 3.2

116 **cannot**

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

118 3.3

119 **conditional**

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

- 122 **3.4**
123 **mandatory**
124 indicates requirements to be followed strictly in order to conform to the document and from which no
125 deviation is permitted
- 126 **3.5**
127 **may**
128 indicates a course of action permissible within the limits of the document
- 129 **3.6**
130 **need not**
131 indicates a course of action permissible within the limits of the document
- 132 **3.7**
133 **optional**
134 indicates a course of action permissible within the limits of the document
- 135 **3.8**
136 **shall**
137 indicates requirements to be followed strictly in order to conform to the document and from which no
138 deviation is permitted
- 139 **3.9**
140 **shall not**
141 indicates requirements to be followed strictly in order to conform to the document and from which no
142 deviation is permitted
- 143 **3.10**
144 **should**
145 indicates that among several possibilities, one is recommended as particularly suitable, without
146 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 147 **3.11**
148 **should not**
149 indicates that a certain possibility or course of action is deprecated but not prohibited

150 **4 Symbols and Abbreviated Terms**

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

- 152 **4.1**
153 **CIM**
154 Common Information Model
- 155 **4.2**
156 **CLP**
157 Command Line Protocol
- 158 **4.3**
159 **DMTF**
160 Distributed Management Task Force

161 **4.4**
162 **IETF**
163 Internet Engineering Task Force

164 **4.5**
165 **SM**
166 Server Management

167 **4.6**
168 **SMI-S**
169 Storage Management Initiative Specification

170 **4.7**
171 **SNIA**
172 Storage Networking Industry Association

173 **4.8**
174 **UFsT**
175 User Friendly Selection Tag

176 **5 Recipes**

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

- 179 • smResetRSC
- 180 • smShowInstance
- 181 • smShowInstances
- 182 • smShowAssociationInstance
- 183 • smShowAssociationInstances
- 184 • smStartRSC
- 185 • smStopRSC

186 **6 Mappings**

187 The following sections detail the mapping of CLP verbs to CIM Operations for each CIM class defined in
188 the [Power Supply Profile](#). Requirements specified here related to the support for a CLP verb for a
189 particular class are solely within the context of this profile.

190 **6.1 CIM_ElementCapabilities**

191 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

192 Table 1 lists each SM CLP verb, the required level of support for the verb in conjunction with the target
193 class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and
194 target. Table 1 is for informational purposes only; in case of a conflict between Table 1 and requirements
195 detailed in the following sections, the text detailed in the following sections supersedes the information in
196 Table 1.

197

Table 1 – 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 6.1.2. |
| Start | Not supported | |
| Stop | Not supported | |

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

200 6.1.1 Ordering of Results

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

- 203 • Results for CIM_ElementCapabilities are unordered; therefore, no algorithm is defined.

204 6.1.2 Show

205 This section describes how to implement the show verb when applied to an instance of
206 CIM_ElementCapabilities. Implementations shall support the use of the show verb with
207 CIM_ElementCapabilities.

208 6.1.2.1 Show Command Form for Multiple Instances Target – 209 CIM_EnabledLogicalElementCapabilities Reference

210 This command form is used to show many instances of CIM_ElementCapabilities. This command form
211 corresponds to a show command issued against instances of CIM_ElementCapabilities where only one
212 reference is specified and the reference is to an instance of CIM_EnabledLogicalElementCapabilities.

213 6.1.2.1.1 Command Form

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

215 6.1.2.1.2 CIM Requirements

216 See CIM_ElementCapabilities in the “CIM Elements” section of the [Power Supply Profile](#) for the list of
217 mandatory properties.

218 6.1.2.1.3 Behavior Requirements

219 6.1.2.1.3.1 Preconditions

220 \$instance represents the instance of CIM_EnabledLogicalElementCapabilities which is referenced by
221 CIM_ElementCapabilities.

222 6.1.2.1.3.2 Pseudo Code

```
223 &smShowAssociationInstances ( "CIM_ElementCapabilities", $instance.getObjectPath() );
224 &smEnd;
```

225 6.1.2.2 Show Command Form for a Single Instance – CIM_PowerSupply Reference

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

229 6.1.2.2.1 Command Form

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

231 6.1.2.2.2 CIM Requirements

232 See CIM_ElementCapabilities in the “CIM Elements” section of the [Power Supply Profile](#) for the list of
 233 mandatory properties.

234 6.1.2.2.3 Behavior Requirements

235 6.1.2.2.3.1 Preconditions

236 `$instance` represents the instance of CIM_PowerSupply which is referenced by
 237 CIM_ElementCapabilities.

238 6.1.2.2.3.2 Pseudo Code

```
239 &smShowAssociationInstances ( "CIM_ElementCapabilities", $instance.getObjectPath() );
240 &smEnd;
```

241 6.1.2.3 Show Command Form for a Single Instance Target – Both References

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

245 6.1.2.3.1 Command Form

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

247 6.1.2.3.2 CIM Requirements

248 See CIM_ElementCapabilities in the “CIM Elements” section of the [Power Supply Profile](#) for the list of
 249 mandatory properties.

250 6.1.2.3.3 Behavior Requirements

251 6.1.2.3.3.1 Preconditions

252 `$instanceA` represents the referenced instance of CIM_PowerSupply through the
 253 CIM_ElementCapabilities association.

254 `$instanceB` represents the instance of CIM_EnabledLogicalElementCapabilities which is referenced by
 255 CIM_ElementCapabilities.

256 **6.1.2.3.4 Pseudo Code**

```

257 &smShowAssociationInstance ( "CIM_ElementCapabilities", $instanceA.getObjectPath(),
258     $instanceB.getObjectPath() );
259 &smEnd;

```

260 **6.2 CIM_EnabledLogicalElementCapabilities**

261 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

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

267 **Table 2 – Command Verb Requirements for CIM_EnabledLogicalElementCapabilities**

| Command Verb | Requirement | Comments |
|--------------|---------------|------------|
| Create | Not supported | |
| Delete | Not supported | |
| Dump | Not supported | |
| Load | Not supported | |
| Reset | Not supported | |
| Set | Not supported | |
| Show | Shall | See 6.2.2. |
| Start | Not supported | |
| Stop | Not supported | |

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

270 **6.2.1 Ordering of Results**

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

- 273 • Results for `CIM_EnabledLogicalElementCapabilities` are unordered; therefore, no algorithm is
 274 defined.

275 **6.2.2 Show**

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

279 **6.2.2.1 Show Command Form for Multiple Instances Target**

280 This command form is used to show many instances of `CIM_EnabledLogicalElementCapabilities`.

281 6.2.2.1.1 Command Form

```
282 show <CIM_EnabledLogicalElementCapabilities multiple instances>
```

283 6.2.2.1.2 CIM Requirements

284 See CIM_EnabledLogicalElementCapabilities in the “CIM Elements” section of the [Power Supply Profile](#)
285 for the list of mandatory properties.

286 6.2.2.1.3 Behavior Requirements

287 6.2.2.1.3.1 Preconditions

288 \$containerInstance represents the instance of CIM_ConcreteCollection with ElementName property
289 that contains “Capabilities” and is associated to the targeted instances of
290 CIM_EnabledLogicalElementCapabilities through the CIM_MemberOfCollection association.

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

292 6.2.2.1.3.2 Pseudo Code

```
293 #propertylist[] = NULL;
294 if ( false == #all)
295     {
296         #propertylist[] = <array of mandatory non-key property names (see CIM
297             Requirements)>;
298     }
299 &smShowInstances ( "CIM_EnabledLogicalElementCapabilities", "CIM_MemberOfCollection",
300     $containerInstance.getObjectPath(), #propertylist[] );
301 &smEnd;
```

302 6.2.2.2 Show Command Form for a Single Instance Target

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

304 6.2.2.2.1 Command Form

```
305 show <CIM_EnabledLogicalElementCapabilities single instance>
```

306 6.2.2.2.2 CIM Requirements

307 See CIM_EnabledLogicalElementCapabilities in the “CIM Elements” section of the [Power Supply Profile](#)
308 for the list of mandatory properties.

309 6.2.2.2.3 Behavior Requirements

310 6.2.2.2.3.1 Preconditions

311 \$instance represents the targeted instance of CIM_EnabledLogicalElementCapabilities.

```
312 $instance=<CIM_EnabledLogicalElementCapabilities single instance>;
```

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

314 **6.2.2.3.2 Pseudo Code**

```

315 #propertylist[] = NULL;
316 if ( false == #all)
317     {
318         #propertylist[] = <array of mandatory non-key property names (see CIM
319             Requirements)>;
320     }
321 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
322 &smEnd;

```

323 **6.3 CIM_IsSpare**

324 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

325 Table 3 lists each SM CLP verb, the required level of support for the verb in conjunction with the target
 326 class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and
 327 target. Table 3 is for informational purposes only; in case of a conflict between Table 3 and requirements
 328 detailed in the following sections, the text detailed in the following sections supersedes the information in
 329 Table 3.

330 **Table 3 – Command Verb Requirements for CIM_IsSpare**

| 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 | |

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

333 **6.3.1 Ordering of Results**

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

- 336 • Results for `CIM_IsSpare` are unordered; therefore, no algorithm is defined.

337 **6.3.2 Show**

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

340 **6.3.2.1 Show Command Form for Multiple Instances Target – CIM_RedundancySet Reference**

341 This command form is used to show many instances of CIM_IsSpare. This command form corresponds to
 342 a `show` command issued against instances of CIM_IsSpare where only one reference is specified and the
 343 reference is to an instance of CIM_RedundancySet

344 **6.3.2.1.1 Command Form**

```
345 show <CIM_IsSpare multiple instances>
```

346 **6.3.2.1.2 CIM Requirements**

347 See CIM_IsSpare in the “CIM Elements” section of the [Power Supply Profile](#) for the list of mandatory
 348 properties.

349 **6.3.2.1.3 Behavior Requirements**

350 **6.3.2.1.3.1 Preconditions**

351 `$instance` represents the instance of CIM_RedundancySet which is referenced by CIM_IsSpare.

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

353 **6.3.2.1.3.2 Pseudo Code**

```
354 #propertylist[] = NULL;
355 if ( false == #all)
356 {
357     #propertylist[] = <array of mandatory non-key property names (see CIM
358     Requirements)>;
359 }
360 &smShowAssociationInstances ( "CIM_IsSpare", $instance.getObjectPath(),
361     #propertylist[] );
362 &smEnd;
```

363 **6.3.2.2 Show Command Form for a Single Instance – CIM_PowerSupply Reference**

364 This command form is used to show a single instance of CIM_IsSpare. This command form corresponds
 365 to a `show` command issued against a single instance of CIM_IsSpare where only one reference is
 366 specified and the reference is to the instance of CIM_PowerSupply.

367 **6.3.2.2.1 Command Form**

```
368 show <CIM_IsSpare single instance>
```

369 **6.3.2.2.2 CIM Requirements**

370 See CIM_IsSpare in the “CIM Elements” section of the [Power Supply Profile](#) for the list of mandatory
 371 properties.

372 **6.3.2.2.3 Behavior Requirements**

373 **6.3.2.2.3.1 Preconditions**

374 `$instance` represents the instance of CIM_PowerSupply which is referenced by CIM_IsSpare.

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

376 **6.3.2.2.3.2 Pseudo Code**

```

377 #propertylist[] = NULL;
378 if ( false == #all)
379 {
380     #propertylist[] = <array of mandatory non-key property names (see CIM
381         Requirements)>;
382 }
383 &smShowAssociationInstances ( "CIM_IsSpare", $instance.getObjectPath(),
384     #propertylist[]);
385 &smEnd;

```

386 **6.3.2.3 Show Command Form for a Single Instance Target – Both References**

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

390 **6.3.2.3.1 Command Form**

```

391 show <CIM_IsSpare single instance>

```

392 **6.3.2.3.2 CIM Requirements**

393 See `CIM_IsSpare` in the “CIM Elements” section of the [Power Supply Profile](#) for the list of mandatory
394 properties.

395 **6.3.2.3.3 Behavior Requirements**396 **6.3.2.3.3.1 Preconditions**

397 `$instanceA` represents the referenced instance of `CIM_PowerSupply` through `CIM_IsSpare` association.

398 `$instanceB` represents the instance of `CIM_RedundancySet` which is referenced by `CIM_IsSpare`.

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

400 **6.3.2.3.3.2 Pseudo Code**

```

401 #propertylist[] = NULL;
402 if ( false == #all)
403 {
404     #propertylist[] = <array of mandatory non-key property names (see CIM
405         Requirements)>;
406 }
407 &smShowAssociationInstance ( "CIM_IsSpare", $instanceA.getObjectPath(),
408     $instanceB.getObjectPath(), #propertylist[] );
409 &smEnd;

```

410 **6.4 CIM_MemberOfCollection**

411 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

412 Table 4 lists each SM CLP verb, the required level of support for the verb in conjunction with the target
413 class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and
414 target. Table 4 is for informational purposes only; in case of a conflict between Table 4 and requirements
415 detailed in the following sections, the text detailed in the following sections supersedes the information in
416 Table 4.

417

Table 4 – Command Verb Requirements for CIM_MemberOfCollection

| 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 | |

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

420 6.4.1 Ordering of Results

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

- 423 • Results for CIM_MemberOfCollection are unordered; therefore, no algorithm is defined.

424 6.4.2 Show

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

428 6.4.2.1 Show Command Form for Multiple Instances Target – CIM_RedundancySet Reference

429 This command form is used to show many instances of CIM_MemberOfCollection. This command form
430 corresponds to a `show` command issued against instances of CIM_MemberOfCollection where only one
431 reference is specified and the reference is to the instance of CIM_RedundancySet.

432 6.4.2.1.1 Command Form

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

434 6.4.2.1.2 CIM Requirements

435 See CIM_MemberOfCollection in the “CIM Elements” section of the [Power Supply Profile](#) for the list of
436 mandatory properties.

437 6.4.2.1.3 Behavior Requirements

438 6.4.2.1.3.1 Preconditions

439 `$instance` represents the instance of CIM_RedundancySet which is referenced by
440 CIM_MemberOfCollection.

441 6.4.2.1.3.2 Pseudo Code

```
442 &smShowAssociationInstances ( "CIM_MemberOfCollection", $instance.getObjectPath() );  
443 &smEnd;
```

444 6.4.2.2 Show Command Form for a Single Instance – CIM_PowerSupply Reference

445 This command form is used to show a single instance of CIM_MemberOfCollection. This command form
446 corresponds to a `show` command issued against a single instance of CIM_MemberOfCollection where
447 only one reference is specified and the reference is to the instance of CIM_PowerSupply.

448 6.4.2.2.1 Command Form

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

450 6.4.2.2.2 CIM Requirements

451 See CIM_MemberOfCollection in the “CIM Elements” section of the [Power Supply Profile](#) for the list of
452 mandatory properties.

453 6.4.2.2.3 Behavior Requirements**454 6.4.2.2.3.1 Preconditions**

455 `$instance` represents the instance of CIM_PowerSupply which is referenced by
456 CIM_MemberOfCollection.

457 6.4.2.2.3.2 Pseudo Code

```
458 &smShowAssociationInstances ( "CIM_MemberOfCollection", $instance.getObjectPath() );  
459 &smEnd;
```

460 6.4.2.3 Show Command Form for a Single Instance Target – Both References

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

464 6.4.2.3.1 Command Form

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

466 6.4.2.3.2 CIM Requirements

467 See CIM_MemberOfCollection in the “CIM Elements” section of the [Power Supply Profile](#) for the list of
468 mandatory properties.

469 6.4.2.3.3 Behavior Requirements**470 6.4.2.3.3.1 Preconditions**

471 `$instanceA` represents the referenced instance of CIM_PowerSupply through
472 CIM_MemberOfCollection association.

473 `$instanceB` represents the instance of CIM_RedundancySet which is referenced by
474 CIM_MemberOfCollection.

475 **6.4.2.3.3.2 Pseudo Code**

```

476 &smShowAssociationInstance ( "CIM_MemberOfCollection", $instanceA.getObjectPath(),
477     $instanceB.getObjectPath() );
478 &smEnd;

```

479 **6.5 CIM_PowerSupply**

480 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

481 Table 5 lists each SM CLP verb, the required level of support for the verb in conjunction with the target
 482 class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and
 483 target. Table 5 is for informational purposes only; in case of a conflict between Table 5 and requirements
 484 detailed in the following sections, the text detailed in the following sections supersedes the information in
 485 Table 5.

486 **Table 5 – Command Verb Requirements for CIM_PowerSupply**

| Command Verb | Requirement | Comments |
|--------------|---------------|------------|
| Create | Not supported | |
| Delete | Not supported | |
| Dump | Not supported | |
| Load | Not supported | |
| Reset | May | See 6.5.2. |
| Set | May | See 6.5.3. |
| Show | Shall | See 6.5.4. |
| Start | May | See 6.5.5. |
| Stop | May | See 6.5.6. |

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

488 **6.5.1 Ordering of Results**

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

- 491 • Results for `CIM_PowerSupply` are unordered; therefore, no algorithm is defined.

492 **6.5.2 Reset**

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

495 **6.5.2.1 Command Form**

```

496 reset <CIM_PowerSupply single instance>

```

497 **6.5.2.2 CIM Requirements**

```

498 uint16 EnabledState;
499 uint16 RequestedState;
500 uint32 CIM_PowerSupply.RequestStateChange (
501     [IN] uint16 RequestedState,
502     [OUT] REF CIM_ConcreteJob Job,
503     [IN] datetime TimeoutPeriod );

```

504 **6.5.2.3 Behavior Requirements**505 **6.5.2.3.1 Preconditions**

506 \$instance represents the targeted instance of CIM_PowerSupply.

```
507 $instance=<CIM_PowerSupply single instance>;
```

508 **6.5.2.3.2 Pseudo Code**

```

509 &smResetRSC ( $instance.getObjectPath() );
510 &smEnd;

```

511 **6.5.3 Set**

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

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

515 **6.5.3.1 General Usage of Set for a Single Property**

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

518 The requirement for supporting modification of a property using this command form shall be equivalent to
519 the requirement for supporting modification of the property using the ModifyInstance operation as defined
520 in the [Power Supply Profile](#).

521 **6.5.3.1.1 Command Form**

```
522 set <CIM_PowerSupply single instance> <propertyname>=<propertyvalue>
```

523 **6.5.3.1.2 CIM Requirements**

524 See CIM_PowerSupply in the “CIM Elements” section of the [Power Supply Profile](#) for the list of mandatory
525 properties.

526 **6.5.3.1.3 Behavior Requirements**

```

527 $instance=<CIM_PowerSupply single instance>
528 #propertyNames[] = {<propertyname>};
529 #propertyValues[] = {<propertyvalue>};
530 &smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
531 &smEnd;

```

532 6.5.3.2 General Usage of Set for Multiple Properties

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

536 The requirement for supporting modification of a property using this command form shall be equivalent to
537 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined
538 in the [Power Supply Profile](#).

539 6.5.3.2.1 Command Form

```
540 set <CIM_PowerSupply single instance> <propertyname1>=<propertyvalue1>  
541 <propertynamen>=<propertyvaluen>
```

542 6.5.3.2.2 CIM Requirements

543 See `CIM_PowerSupply` in the “CIM Elements” section of the [Power Supply Profile](#) for the list of mandatory
544 properties.

545 6.5.3.2.3 Behavior Requirements

```
546 $instance=<CIM_PowerSupply single instance>  
547 #propertyName[] = {<propertyname>};  
548 for #i < n  
549 {  
550     #propertyName[#i] = <propertyname#i>  
551     #propertyValue[#i] = <propertyvalue#i>  
552 }  
553 &smSetInstance ( $instance, #propertyName[], #propertyValue[] );  
554 &smEnd;
```

555 6.5.3.3 Set RequestedState to “Offline”

556 This section describes how to change the state of the power supply represented by `CIM_PowerSupply` to
557 “Enabled but Offline”.

558 6.5.3.3.1 Command Form

```
559 set <CIM_PowerSupply single instance> RequestedState="Offline"
```

560 6.5.3.3.2 CIM Requirements

561 See `CIM_PowerSupply` in the “CIM Elements” section of the [Power Supply Profile](#) for the list of mandatory
562 properties.

563 6.5.3.3.3 Behavior Requirements

564 6.5.3.3.3.1 Preconditions

565 `$instance` represents the targeted instance of `CIM_PowerSupply`.

```
566 $instance=<CIM_PowerSupply single instance>;
```

567 6.5.3.3.3.2 Pseudo Code

```
568 //“Offline” is valuemap 6  
569 &smRequestStateChange ( $instance.getObjectPath(), 6 );  
570 &smEnd;
```

571 6.5.4 Show

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

574 6.5.4.1 Show Command Form for Multiple Instances Target

575 This command form is used to show many instances of `CIM_PowerSupply`.

576 6.5.4.1.1 Command Form

```
577 show <CIM_PowerSupply multiple instances>
```

578 6.5.4.1.2 CIM Requirements

579 See `CIM_PowerSupply` in the “CIM Elements” section of the [Power Supply Profile](#) for the list of mandatory
580 properties.

581 6.5.4.1.3 Behavior Requirements

582 6.5.4.1.3.1 Preconditions

583 `$containerInstance` represents the instance of `CIM_ComputerSystem` which represents the
584 container system and is associated to the targeted instances of `CIM_PowerSupply` through the
585 `CIM_SystemDevice` association.

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

587 6.5.4.1.3.2 Pseudo Code

```
588 #propertylist[] = NULL;  
589 if ( false == #all)  
590 {  
591     #propertylist[] = <array of mandatory non-key property names (see CIM  
592         Requirements)>;  
593 }  
594 &smShowInstances ( "CIM_PowerSupply", "CIM_SystemDevice",  
595     $containerInstance.getObjectPath(), #propertylist[] );  
596 &smEnd;
```

597 6.5.4.2 Show Command Form for a Single Instance Target

598 This command form is used to show a single instance of `CIM_PowerSupply`.

599 6.5.4.2.1 Command Form

```
600 show <CIM_PowerSupply single instance>
```

601 6.5.4.2.2 CIM Requirements

602 See `CIM_PowerSupply` in the “CIM Elements” section of the [Power Supply Profile](#) for the list of mandatory
603 properties.

604 **6.5.4.2.3 Behavior Requirements**

605 **6.5.4.2.3.1 Preconditions**

606 `$instance` represents the targeted instance of `CIM_PowerSupply`.

```
607 $instance=<CIM_PowerSupply single instance>;
```

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

609 **6.5.4.2.3.2 Pseudo Code**

```
610 #propertylist[] = NULL;
611 if ( false == #all)
612 {
613     #propertylist[] = <array of mandatory non-key property names (see CIM
614         Requirements)>;
615 }
616 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
617 &smEnd;
```

618 **6.5.5 Start**

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

621 **6.5.5.1.1 Command Form**

```
622 start <CIM_PowerSupply single instance>
```

623 **6.5.5.1.2 CIM Requirements**

```
624 uint16 EnabledState;
625 uint16 RequestedState;
626 uint32 CIM_PowerSupply.RequestStateChange (
627     [IN] uint16 RequestedState,
628     [OUT] REF CIM_ConcreteJob Job,
629     [IN] datetime TimeoutPeriod );
```

630 **6.5.5.1.3 Behavior Requirements**

631 **6.5.5.1.3.1 Preconditions**

632 `$instance` represents the targeted instance of `CIM_PowerSupply`.

```
633 $instance=<CIM_PowerSupply single instance>;
```

634 **6.5.5.1.3.2 Pseudo Code**

```
635 &smStartRSC ( $instance.getObjectPath() );
636 &smEnd;
```

637 **6.5.6 Stop**

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

640 **6.5.6.1.1 Command Form**641 `stop <CIM_PowerSupply single instance>`642 **6.5.6.1.2 CIM Requirements**

```

643 uint16 EnabledState;
644 uint16 RequestedState;
645 uint32 CIM_PowerSupply.RequestStateChange (
646     [IN] uint16 RequestedState,
647     [OUT] REF CIM_ConcreteJob Job,
648     [IN] datetime TimeoutPeriod );

```

649 **6.5.6.1.3 Behavior Requirements**650 **6.5.6.1.3.1 Preconditions**651 `$instance` represents the targeted instance of `CIM_PowerSupply`.652 `$instance=<CIM_PowerSupply single instance>;`653 **6.5.6.1.3.2 Pseudo Code**

```

654 &smStopRSC ( $instance.getObjectPath() );
655 &smEnd;

```

656 **6.6 CIM_RedundancySet**657 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

658 Table 6 lists each SM CLP verb, the required level of support for the verb in conjunction with the target
 659 class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and
 660 target. Table 6 is for informational purposes only; in case of a conflict between Table 6 and requirements
 661 detailed in the following sections, the text detailed in the following sections supersedes the information in
 662 Table 6.

663 **Table 6 – Command Verb Requirements for CIM_RedundancySet**

| Command Verb | Requirement | Comments |
|--------------|---------------|------------|
| Create | Not supported | |
| Delete | Not supported | |
| Dump | Not supported | |
| Load | Not supported | |
| Reset | Not supported | |
| Set | May | See 6.6.2. |
| Show | Shall | See 6.6.3. |
| Start | Not supported | |
| Stop | Not supported | |

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

665 6.6.1 Ordering of Results

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

- 668 • Results for CIM_RedundancySet are unordered; therefore, no algorithm is defined.

669 6.6.2 Set

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

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

673 6.6.2.1 Set Command for Failovers

674 This section describes how to use the `set` verb when it is applied to an instance of CIM_RedundancySet
675 to failover from an active power supply to a spare power supply.

676 6.6.2.1.1 Command Form

```
677 set <CIM_RedundancySet single instance> failoverfrom=<CIM_PowerSupply single instance>
678     failovertto=<CIM_PowerSupply single instance>
```

679 6.6.2.1.2 CIM Requirements

```
680 uint32 CIM_RedundancySet.Failover (
681     [IN] REF CIM_ManagedElement FailoverFrom,
682     [IN] REF CIM_ManagedElement FailoverTo);
```

683 6.6.2.1.3 Behavior Requirements

684 6.6.2.1.3.1 Preconditions

```
685 $instance=<CIM_RedundancySet single instance>
686 $FailoverFrom=<failoverfrom requested instance of CIM_PowerSupply>
687 $FailoverTo=<failovertto requested instance of CIM_PowerSupply>
```

688 6.6.2.1.3.2 Pseudo Code

```
689 %InArguments[] = { newArgument ( "FailoverFrom",
690     $FailoverFrom.getObjectPath() ), newArgument ( "FailoverTo",
691     $FailoverTo.getObjectPath () ) };
692 %OutArguments[] = {};
693 #Error = InvokeMethod ($target->,
694     "Failover",
695     %InArguments[],
696     %OutArguments[],
697     #returnStatus);
698 if (0 != #Error.code)
699 {
700     //method invocation failed
701     if ( (null != #Error.$error) && (null != #Error.$error[0]) )
702     {
703         //if the method invocation contains an embedded error
704         //use it for the Error for the overall job
705         &smAddError($job, #Error.$error[0]);
```

```

706     &smMakeCommandStatus($job);
707     &smEnd;
708 }
709 else if ( 17 == #Error.code ) {
710     //17 - CIM_ERR_METHOD_NOT_FOUND
711     // The specified extrinsic method does not exist.
712     $OperationError = smNewInstance("CIM_Error");
713     // CIM_ERR_METHOD_NOT_FOUND
714     $OperationError.CIMStatusCode = 17;
715     //Software Error
716     $OperationError.ErrorType = 10;
717     //Unknown
718     $OperationError.PerceivedSeverity = 0;
719     $OperationError.OwningEntity = DMTF:SMCLP;
720     $OperationError.MessageID = 0x00000001;
721     $OperationError.Message = "Operation is not supported."
722     &smAddError($job, $OperationError);
723     &smMakeCommandStatus($job);
724     &smEnd;
725 }
726 else
727 {
728     //operation failed, but no detailed error instance, need to make one up
729     //make an Error instance and associate with job for Operation
730     $OperationError = smNewInstance("CIM_Error");
731     //CIM_ERR_FAILED
732     $OperationError.CIMStatusCode = 1;
733     //Software Error
734     $OperationError.ErrorType = 4;
735     //Unknown
736     $OperationError.PerceivedSeverity = 0;
737     $OperationError.OwningEntity = DMTF:SMCLP;
738     $OperationError.MessageID = 0x00000009;
739     $OperationError.Message = "An internal software error has occurred.";
740     &smAddError($job, $OperationError);
741     &smMakeCommandStatus($job);
742     &smEnd;
743 }
744 }//if CIM op failed
745 else if (0 == #returnStatus) {
746     //completed successfully
747     &smCommandCompleted($job);
748     &smEnd;
749 }
750 else if (1 == #returnStatus) {
751     //unsupported
752     $OperationError = smNewInstance("CIM_Error");
753     //CIM_ERR_NOT_SUPPORTED
754     $OperationError.CIMStatusCode = 7;

```

```

755 //Other
756 $OperationError.ErrorType = 1;
757 //Low
758 $OperationError.PerceivedSeverity = 2;
759 $OperationError.OwningEntity = DMTF:SMCLP;
760 $OperationError.MessageID = 0x00000001;
761 $OperationError.Message = "Operation is not supported.";
762 &smAddError($job, $OperationError);
763 &smMakeCommandStatus($job);
764 &smEnd;
765 }
766 else if (2 == #returnStatus) {
767 //generic failure
768 $OperationError = smNewInstance("CIM_Error");
769 //CIM_ERR_FAILED
770 $OperationError.CIMStatusCode = 1;
771 //Other
772 $OperationError.ErrorType = 1;
773 //Low
774 $OperationError.PerceivedSeverity = 2;
775 $OperationError.OwningEntity = DMTF:SMCLP;
776 $OperationError.MessageID = 0x00000002;
777 $OperationError.Message = "Failed. No further information is available.";
778 &smAddError($job, $OperationError);
779 &smMakeCommandStatus($job);
780 }
781 else {
782 //unspecified return code, generic failure
783 $OperationError = smNewInstance("CIM_Error");
784 //CIM_ERR_FAILED
785 $OperationError.CIMStatusCode = 1;
786 //Other
787 $OperationError.ErrorType = 1;
788 //Low
789 $OperationError.PerceivedSeverity = 2;
790 $OperationError.OwningEntity = DMTF:SMCLP;
791 $OperationError.MessageID = 0x00000002;
792 $OperationError.Message = "Failed. No further information is available.";
793 &smAddError($job, $OperationError);
794 &smMakeCommandStatus($job);
795 &smEnd;
796 }

```

797 6.6.2.2 General Usage of Set for a Single Property

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

800 The requirement for supporting modification of a property using this command form shall be equivalent to
 801 the requirement for supporting modification of the property using the ModifyInstance operation as defined
 802 in the [Power Supply Profile](#).

803 6.6.2.2.1 Command Form

```
804 set <CIM_RedundancySet single instance> <propertyname>=<propertyvalue>
```

805 6.6.2.2.2 CIM Requirements

806 See CIM_RedundancySet in the “CIM Elements” section of the [Power Supply Profile](#) for the list of
 807 mandatory properties.

808 6.6.2.2.3 Behavior Requirements

```
809 $instance=<CIM_RedundancySet single instance>
810 #propertyNames[] = {<propertyname>};
811 #propertyValues[] = {<propertyvalue>};
812 &smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
813 &smEnd;
```

814 6.6.2.3 General Usage of Set for Multiple Properties

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

818 The requirement for supporting modification of a property using this command form shall be equivalent to
 819 the requirement for supporting modification of the property using the ModifyInstance operation as defined
 820 in the [Power Supply Profile](#).

821 6.6.2.3.1 Command Form

```
822 set <CIM_RedundancySet single instance> <propertyname1>=<propertyvalue1>
823 <propertynamen>=<propertyvaluen>
```

824 6.6.2.3.2 CIM Requirements

825 See CIM_RedundancySet in the “CIM Elements” section of the [Power Supply Profile](#) for the list of
 826 mandatory properties.

827 6.6.2.3.3 Behavior Requirements

```
828 $instance=<CIM_RedundancySet single instance>
829 #propertyNames[] = {<propertyname>};
830 for #i < n
831 {
832     #propertyNames[#i] = <propertyname#i>
833     #propertyValues[#i] = <propertyvalue#i>
834 }
835 &smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
836 &smEnd;
```

837 6.6.3 Show

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

840 **6.6.3.1 Show Command Form for Multiple Instances Target**

841 This command form is used to show many instances of CIM_RedundancySet.

842 **6.6.3.1.1 Command Form**

```
843 show <CIM_RedundancySet multiple instances>
```

844 **6.6.3.1.2 CIM Requirements**

845 See CIM_RedundancySet in the “CIM Elements” section of the [Power Supply Profile](#) for the list of
846 mandatory properties.

847 **6.6.3.1.3 Behavior Requirements**

848 **6.6.3.1.3.1 Preconditions**

849 \$containerInstance represents the instance of CIM_ComputerSystem which represents the
850 container system and is associated to the targeted instances of CIM_RedundancySet through the
851 CIM_OwningCollectionElement association.

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

853 **6.6.3.1.3.2 Pseudo Code**

```
854 #propertylist[] = NULL;
855 if ( false == #all)
856     {
857         #propertylist[] = <array of mandatory non-key property names (see CIM
858             Requirements)>;
859     }
860 &smShowInstances ( "CIM_RedundancySet", "CIM_OwningCollectionElement",
861     $containerInstance.getObjectPath(), #propertylist[] );
862 &smEnd;
```

863 **6.6.3.2 Show Command Form for a Single Instance Target**

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

865 **6.6.3.2.1 Command Form**

```
866 show <CIM_RedundancySet single instance>
```

867 **6.6.3.2.2 CIM Requirements**

868 See CIM_RedundancySet in the “CIM Elements” section of the [Power Supply Profile](#) for the list of
869 mandatory properties.

870 **6.6.3.2.3 Behavior Requirements**

871 **6.6.3.2.3.1 Preconditions**

872 \$instance represents the targeted instance of CIM_RedundancySet.

```
873 $instance=<CIM_RedundancySet single instance>;
```

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

875 **6.6.3.2.3.2 Pseudo Code**

```

876 #propertylist[] = NULL;
877 if ( false == #all)
878     {
879     #propertylist[] = <array of mandatory non-key property names (see CIM
880     Requirements)>;
881     }
882 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
883 &smEnd;

```

884 **6.7 CIM_SystemDevice**

885 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

886 Table 7 lists each SM CLP verb, the required level of support for the verb in conjunction with the target
 887 class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and
 888 target. Table 7 is for informational purposes only; in case of a conflict between Table 7 and requirements
 889 detailed in the following sections, the text detailed in the following sections supersedes the information in
 890 Table 7.

891 **Table 7 – Command Verb Requirements for CIM_SystemDevice**

| Command Verb | Requirement | Comments |
|--------------|---------------|------------|
| Create | Not supported | |
| Delete | Not supported | |
| Dump | Not supported | |
| Load | Not supported | |
| Reset | Not supported | |
| Set | Not supported | |
| Show | Shall | See 6.7.2. |
| Start | Not supported | |
| Stop | Not supported | |

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

894 **6.7.1 Ordering of Results**

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

- 897 • Results for `CIM_SystemDevice` are unordered; therefore, no algorithm is defined.

898 **6.7.2 Show**

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

901 **6.7.2.1 Show Command Form for Multiple Instances Target – CIM_ComputerSystem Reference**

902 This command form is used to show many instances of CIM_SystemDevice. This command form
903 corresponds to a `show` command issued against the instance of CIM_SystemDevice where only one
904 reference is specified and the reference is to the scoping instance of CIM_ComputerSystem.

905 **6.7.2.1.1 Command Form**

```
906 show <CIM_SystemDevice multiple instances>
```

907 **6.7.2.1.2 CIM Requirements**

908 See CIM_SystemDevice in the “CIM Elements” section of the [Power Supply Profile](#) for the list of
909 mandatory properties.

910 **6.7.2.1.3 Behavior Requirements**

911 **6.7.2.1.3.1 Preconditions**

912 `$instance` represents the instance of a CIM_ComputerSystem, which is referenced by
913 CIM_SystemDevice.

914 **6.7.2.1.3.2 Pseudo Code**

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

917 **6.7.2.2 Show Command Form for a Single Instance Target – CIM_PowerSupply Reference**

918 This command form is used to show a single instance of CIM_SystemDevice. This command form
919 corresponds to a `show` command issued against a single instance of CIM_SystemDevice, where only one
920 reference is specified and the reference is to the instance of CIM_PowerSupply.

921 **6.7.2.2.1 Command Form**

```
922 show <CIM_SystemDevice single instance>
```

923 **6.7.2.2.2 CIM Requirements**

924 See CIM_SystemDevice in the “CIM Elements” section of the [Power Supply Profile](#) for the list of
925 mandatory properties.

926 **6.7.2.2.3 Behavior Requirements**

927 **6.7.2.2.3.1 Preconditions**

928 `$instance` represents the instance of CIM_PowerSupply which is referenced by CIM_SystemDevice.

929 **6.7.2.2.3.2 Pseudo Code**

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

932 **6.7.2.3 Show Command Form for a Single Instance Target – Both References**

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

936 **6.7.2.3.1 Command Form**937 `show <CIM_SystemDevice single instance>`938 **6.7.2.3.2 CIM Requirements**939 See CIM_SystemDevice in the “CIM Elements” section of the [Power Supply Profile](#) for the list of
940 mandatory properties.941 **6.7.2.3.3 Behavior Requirements**942 **6.7.2.3.3.1 Preconditions**943 \$instanceA represents the referenced instance of CIM_PowerSupply through CIM_SystemDevice
944 association.945 \$instanceB represents the instance of CIM_ComputerSystem which is referenced by
946 CIM_SystemDevice.947 **6.7.2.3.3.2 Pseudo Code**948 `&smShowAssociationInstance ("CIM_SystemDevice", $instanceA.getObjectPath(),`
949 `$instanceB.getObjectPath());`
950 `&smEnd;`951 **6.8 CIM_OwningCollectionElement**952 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).953 Table 8 lists each SM CLP verb, the required level of support for the verb in conjunction with the target
954 class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and
955 target. Table 8 is for informational purposes only; in case of a conflict between Table 8 and requirements
956 detailed in the following sections, the text detailed in the following sections supersedes the information in
957 Table 8.958 **Table 8 – Command Verb Requirements for CIM_OwningCollectionElement**

| 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 | |

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

961 **6.8.1 Ordering of Results**

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

- 964 • Results for CIM_OwningCollectionElement are unordered; therefore, no algorithm is defined.

965 **6.8.2 Show**

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

969 **6.8.2.1 Show Command Form for Multiple Instances Target – CIM_ComputerSystem Reference**

970 This command form is used to show many instances of CIM_OwningCollectionElement. This command
971 form corresponds to a `show` command issued against the instance of CIM_OwningCollectionElement
972 where only one reference is specified and the reference is to the scoping instance of
973 CIM_ComputerSystem.

974 **6.8.2.1.1 Command Form**

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

976 **6.8.2.1.2 CIM Requirements**

977 See CIM_OwningCollectionElement in the “CIM Elements” section of the [Power Supply Profile](#) for the list
978 of mandatory properties.

979 **6.8.2.1.3 Behavior Requirements**

980 **6.8.2.1.3.1 Preconditions**

981 `$instance` represents the instance of a CIM_ComputerSystem, which is referenced by
982 CIM_OwningCollectionElement.

983 **6.8.2.1.3.2 Pseudo Code**

```
984 &smShowAssociationInstances ( "CIM_OwningCollectionElement",  
985     $instance.getObjectPath() );  
986 &smEnd;
```

987 **6.8.2.2 Show Command Form for a Single Instance Target – CIM_RedundancySet Reference**

988 This command form is used to show a single instance of CIM_OwningCollectionElement. This command
989 form corresponds to a `show` command issued against a single instance of
990 CIM_OwningCollectionElement, where only one reference is specified and the reference is to an instance
991 of CIM_RedundancySet.

992 **6.8.2.2.1 Command Form**

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

994 **6.8.2.2.2 CIM Requirements**

995 See CIM_OwningCollectionElement in the “CIM Elements” section of the [Power Supply Profile](#) for the list
996 of mandatory properties.

997 6.8.2.2.3 Behavior Requirements**998 6.8.2.2.3.1 Preconditions**

999 \$instance represents the instance of CIM_RedundancySet which is referenced by
1000 CIM_OwningCollectionElement.

1001 6.8.2.2.3.2 Pseudo Code

```
1002 &smShowAssociationInstances ( "CIM_OwningCollectionElement",  
1003     $instance.getObjectPath() );  
1004 &smEnd;
```

1005 6.8.2.3 Show Command Form for a Single Instance Target – Both References

1006 This command form is for the `show` verb applied to a single instance. This command form corresponds to
1007 a `show` command issued against `CIM_OwningCollectionElement` where both references are specified
1008 and therefore the desired instance is unambiguously identified.

1009 6.8.2.3.1 Command Form

```
1010 show <CIM_OwningCollectionElement single instance>
```

1011 6.8.2.3.2 CIM Requirements

1012 See `CIM_OwningCollectionElement` in the “CIM Elements” section of the [Power Supply Profile](#) for the list
1013 of mandatory properties.

1014 6.8.2.3.3 Behavior Requirements**1015 6.8.2.3.3.1 Preconditions**

1016 \$instanceA represents the referenced instance of `CIM_RedundancySet` through
1017 `CIM_OwningCollectionElement` association.

1018 \$instanceB represents the instance of `CIM_ComputerSystem` which is referenced by
1019 `CIM_OwningCollectionElement`.

1020 6.8.2.3.3.2 Pseudo Code

```
1021 &smShowAssociationInstance ( "CIM_OwningCollectionElement",  
1022     $instanceA.getObjectPath(), $instanceB.getObjectPath() );  
1023 &smEnd;
```

1024

**ANNEX A
(informative)**

Change Log

1025
1026
1027
1028
1029

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

1030