



1
2
3
4

Document Number: DSP0809

Date: 2009-06-04

Version: 1.0.0

5 **System Memory Profile SM CLP 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27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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_ElementCapabilities	9
45	6.2 CIM_EnabledLogicalElementCapabilities.....	11
46	6.3 CIM_Memory	13
47	6.4 CIM_SystemDevice	16
48	ANNEX A (informative) Change Log	20
49		

50 Tables

51	Table 1 – Command Verb Requirements for CIM_ElementCapabilities	9
52	Table 2 – Command Verb Requirements for CIM_EnabledLogicalElementCapabilities.....	12
53	Table 3 – Command Verb Requirements for CIM_Memory	14
54	Table 4 – Command Verb Requirements for CIM_SystemDevice	17
55		

57

Foreword

58 The *System Memory Profile SM CLP Mapping Specification* (DSP0809) was prepared by the Server
59 Management Working Group.

60 **Conventions**

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

63 **Acknowledgements**

64 The authors wish to acknowledge the following participants from the DTMF Server Management Working
65 Group:

- 66 • Khachatur Papanyan – Dell Inc.
- 67 • Jon Hass – Dell Inc.
- 68 • Jeff Hilland – HP
- 69 • Christina Shaw – HP
- 70 • Aaron Merkin – IBM
- 71 • Jeff Lynch – IBM
- 72 • Perry Vincent – Intel
- 73 • John Leung – Intel

74

75

Introduction

76 This document defines the SM CLP mapping for CIM elements described in the [System Memory Profile](#).
77 The information in this specification, combined with the [SM CLP-to-CIM Common Mapping Specification](#)
78 [1.0](#), is intended to be sufficient to implement SM CLP commands relevant to the classes, properties and
79 methods described in [System Memory Profile](#) using CIM operations.

80 The target audience for this specification is implementers of the SM CLP support for the [System Memory](#)
81 [Profile](#).

82 System Memory Profile SM CLP Mapping Specification

83 1 Scope

84 This specification contains the requirements for an implementation of the SM CLP to provide access to
85 and implement the behaviors of the [System Memory Profile](#).

86 2 Normative References

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

90 2.1 Approved References

91 DMTF DSP1026, *System Memory Profile 1.0*,
92 http://www.dmtf.org/standards/published_documents/DSP1026_1.0.pdf

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

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

97 2.2 Other References

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

100 3 Terms and Definitions

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

102 3.1

103 **can**

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

105 3.2

106 **cannot**

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

108 3.3

109 **conditional**

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

112 3.4

113 **mandatory**

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

- 116 **3.5**
117 **may**
118 indicates a course of action permissible within the limits of the document
- 119 **3.6**
120 **need not**
121 indicates a course of action permissible within the limits of the document
- 122 **3.7**
123 **optional**
124 indicates a course of action permissible within the limits of the document
- 125 **3.8**
126 **shall**
127 indicates requirements to be followed strictly in order to conform to the document and from which no
128 deviation is permitted
- 129 **3.9**
130 **shall not**
131 indicates requirements to be followed strictly in order to conform to the document and from which no
132 deviation is permitted
- 133 **3.10**
134 **should**
135 indicates that among several possibilities, one is recommended as particularly suitable, without
136 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 137 **3.11**
138 **should not**
139 indicates that a certain possibility or course of action is deprecated but not prohibited

140 **4 Symbols and Abbreviated Terms**

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

- 142 **4.1**
143 **CIM**
144 Common Information Model
- 145 **4.2**
146 **CLP**
147 Command Line Protocol
- 148 **4.3**
149 **DMTF**
150 Distributed Management Task Force
- 151 **4.4**
152 **IETF**
153 Internet Engineering Task Force

154 **4.5**
 155 **SM**
 156 Server Management

157 **4.6**
 158 **SMI**
 159 Storage Management Initiative

160 **4.7**
 161 **SNIA**
 162 Storage Networking Industry Association

163 **5 Recipes**

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

- 166 • smShowInstance
- 167 • smShowInstances
- 168 • smShowAssociationInstance
- 169 • smShowAssociationInstances

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

171 **6 Mappings**

172 The following sections detail the mapping of CLP verbs to CIM Operations for each CIM class defined in
 173 the [System Memory Profile](#).

174 **6.1 CIM_ElementCapabilities**

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

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

181 **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	

Command Verb	Requirement	Comments
Show	Shall	See 6.1.2
Start	Not supported	
Stop	Not supported	

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

184 6.1.1 Ordering of Results

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

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

188 6.1.2 Show

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

192 6.1.2.1 Show Command Form for a Single Instance Target – CIM_Memory Reference

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

196 6.1.2.1.1 Command Form

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

198 6.1.2.1.2 CIM Requirements

199 See CIM_ElementCapabilities in the “CIM Elements” section of the [System Memory Profile](#) for the list of
200 mandatory properties.

201 6.1.2.1.3 Behavior Requirements

202 6.1.2.1.3.1 Preconditions

203 In this section `$instance` represents the instance of CIM_Memory which is referenced by
204 CIM_ElementCapabilities.

205 6.1.2.1.3.2 Pseudo Code

```
206 &smShowAssociationInstances ( "CIM_ElementCapabilities",  
207     $instance.getInstancePath() );  
208 &smEnd;
```

209 6.1.2.2 Show Command Form for Multiple Instances –CIM_EnabledLogicalElementCapabilities 210 Reference

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

215 6.1.2.2.1 Command Form

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

217 6.1.2.2.2 CIM Requirements

218 See CIM_ElementCapabilities in the “CIM Elements” section of the [System Memory Profile](#) for the list of
219 mandatory properties and CIM classes that can be referenced.

220 6.1.2.2.3 Behavior Requirements

221 6.1.2.2.3.1 Preconditions

222 In this section \$instance represents the instance of CIM_EnabledLogicalElementCapabilities which is
223 referenced by CIM_ElementCapabilities.

224 6.1.2.2.3.2 Pseudo Code

```
225 &smShowAssociationInstances ( "CIM_ElementCapabilities",  
226     $instance.getInstancePath() );  
227 &smEnd;
```

228 6.1.2.3 Show Command Form for a Single Instance – Both References

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

232 6.1.2.3.1 Command Form

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

234 6.1.2.3.2 CIM Requirements

235 See CIM_ElementCapabilities in the “CIM Elements” section of the [System Memory Profile](#) for the list of
236 mandatory properties and CIM classes that can be referenced.

237 6.1.2.3.3 Behavior Requirements

238 6.1.2.3.3.1 Preconditions

239 In this section \$instanceA represents the referenced instance of CIM_Memory through
240 CIM_ElementCapabilities association. \$instanceB represents the instance of
241 CIM_EnabledLogicalElementCapabilities which is referenced by CIM_ElementCapabilities.

242 6.1.2.3.3.2 Pseudo Code

```
243 &smShowAssociationInstance ( "CIM_ElementCapabilities", $instanceA.getInstancePath(),  
244     $instanceB.getInstancePath() );  
245 &smEnd;
```

246 6.2 CIM_EnabledLogicalElementCapabilities

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

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

251 detailed in the following sections, the text detailed in the following sections supersedes the information in
252 Table 2.

253 **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	

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

256 6.2.1 Ordering of Results

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

- 259 • Results for CIM_EnabledLogicalElementCapabilities are unordered; therefore, no algorithm is
260 defined.

261 6.2.2 Show

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

265 6.2.2.1 Show Command Form for Multiple Instances Target

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

267 6.2.2.1.1 Command Form

```
268 show <CIM_EnabledLogicalElementCapabilities multiple instances>
```

269 6.2.2.1.2 CIM Requirements

270 See CIM_EnabledLogicalElementCapabilities in the “CIM Elements” section of the [System Memory](#)
271 [Profile](#) for the list of mandatory properties.

272 6.2.2.1.3 Behavior Requirements

273 6.2.2.1.3.1 Preconditions

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

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

278 6.2.2.1.3.2 Pseudo Code

```
279 #propertylist[] = NULL;
280 if ( false == #all) {
281     #propertylist[] = <array of mandatory non-key property names (see CIM
282         Requirements)>;
283 }
284 &smShowInstances ( "CIM_EnabledLogicalElementCapabilities", "CIM_MemberOfCollection",
285     $containerInstance.getInstancePath(), #propertylist[] );
286 &smEnd;
```

287 6.2.2.2 Show Command Form for a Single Instance Target

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

289 6.2.2.2.1 Command Form

```
290 show <CIM_EnabledLogicalElementCapabilities single instance>
```

291 6.2.2.2.2 CIM Requirements

292 See CIM_EnabledLogicalElementCapabilities in the “CIM Elements” section of the [System Memory Profile](#) for the list of mandatory properties.

294 6.2.2.2.3 Behavior Requirements

295 6.2.2.2.3.1 Preconditions

296 In this section \$instance represents the targeted instance of CIM_EnabledLogicalElementCapabilities.

```
297 $instance=<CIM_EnabledLogicalElementCapabilities single instance>;
```

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

299 6.2.2.2.3.2 Pseudo Code

```
300 #propertylist[] = NULL;
301 if ( false == #all) {
302     #propertylist[] = <array of mandatory non-key property names (see CIM
303         Requirements)>;
304 }
305 &smShowInstance ( $instance, #propertylist[] );
306 &smEnd;
```

307 6.3 CIM_Memory

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

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

314

Table 3 – Command Verb Requirements for CIM_Memory

Command Verb	Requirement	Comments
Create	Shall not	
Delete	Shall not	
Dump	Shall not	
Load	Shall not	
Reset	Shall not	
Set	May	See 6.3.2.
Show	Shall	See 6.3.3.
Start	Shall not	
Stop	Shall not	

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

317 6.3.1 Ordering of Results

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

- 320 • Results for CIM_Memory are unordered; therefore, no algorithm is defined.

321 6.3.2 Set

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

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

325 6.3.2.1 General Usage of Set for a Single Property

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

328 The requirement for supporting modification of a property using this command form shall be equivalent to
329 the requirement for supporting modification of the property using the ModifyInstance operation as defined
330 in the [System Memory Profile](#).

331 6.3.2.1.1 Command Form

```
332 set <CIM_Memory single instance> <propertyname>=<propertyvalue>
```

333 6.3.2.1.2 CIM Requirements

334 The ModifyInstance operation is required.

335 6.3.2.1.3 Behavior Requirements

```
336 $instance=<CIM_Memory single instance>
337 #propertyName[] = {<propertyname>};
338 #propertyValues[] = {<propertyvalue>};
339 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
340 &smEnd;
```

341 6.3.2.2 General Usage of Set for Multiple Properties

342 This command form corresponds to the general usage of the set verb to modify multiple properties of a
343 target instance where there isn't an explicit relationship between the properties. This is the most common
344 case.

345 The requirement for supporting modification of a property using this command form shall be equivalent to
346 the requirement for supporting modification of the property using the ModifyInstance operation as defined
347 in the [System Memory Profile](#).

348 6.3.2.2.1 Command Form

```
349 set <CIM_Memory single instance> <propertyname1>=<propertyvalue1>  
350 <propertynamen>=<propertyvaluen>
```

351 6.3.2.2.2 CIM Requirements

352 The ModifyInstance operation is required.

353 6.3.2.2.3 Behavior Requirements

```
354 $instance=<CIM_Memory single instance>  
355 #propertyName[] = {<propertyname>};  
356 for #i < n  
357 {  
358     #propertyName[#i] = <propertyname#i>  
359     #propertyValue[#i] = <propertyvalue#i>  
360 }  
361 &smSetInstance ( $instance, #propertyName[], #propertyValue[] );  
362 &smEnd;
```

363 6.3.3 Show

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

366 6.3.3.1 Show Command Form for Multiple Instances Target

367 This command form is used to show many instances of CIM_Memory.

368 6.3.3.1.1 Command Form

```
369 show <CIM_Memory multiple instances>
```

370 6.3.3.1.2 CIM Requirements

371 See CIM_Memory in the "CIM Elements" section of the [System Memory Profile](#) for the list of mandatory
372 properties.

373 6.3.3.1.3 Behavior Requirements

374 6.3.3.1.3.1 Preconditions

375 In this section \$containerInstance represents the instance of CIM_ComputerSystem which
376 represents the container system and is associated to the targeted instances of CIM_Memory through the
377 CIM_SystemDevice association.

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

379 **6.3.3.1.3.2 Pseudo Code**

```

380 #propertylist[] = NULL;
381 if ( false == #all) {
382     #propertylist[] = <array of mandatory non-key property names (see CIM
383         Requirements)>;
384 }
385 &smShowInstances ( "CIM_Memory", "CIM_SystemDevice",
386     $containerInstance.getObjectPath() );
387 &smEnd;

```

388 **6.3.3.2 Show Command Form for a Single Instance Target**

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

390 **6.3.3.2.1 Command Form**

```

391 show <CIM_Memory single instance>

```

392 **6.3.3.2.2 CIM Requirements**

393 See CIM_Memory in the "CIM Elements" section of the [System Memory Profile](#) for the list of mandatory
394 properties.

395 **6.3.3.2.3 Behavior Requirements**396 **6.3.3.2.3.1 Preconditions**

397 In this section \$instance represents the targeted instance of CIM_Memory.

```

398 $instance=<CIM_Memory single instance>;

```

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

400 **6.3.3.2.3.2 Pseudo Code**

```

401 #propertylist[] = NULL;
402 if ( false == #all) {
403     #propertylist[] = <array of mandatory non-key property names (see CIM
404         Requirements)>;
405 }
406 &smShowInstance ( $instance );
407 &smEnd;

```

408 **6.4 CIM_SystemDevice**

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

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

415

Table 4 – Command Verb Requirements for CIM_SystemDevice

Command Verb	Requirement	Comments
Create	Shall not	
Delete	Shall not	
Dump	Shall not	
Load	Shall not	
Reset	Shall not	
Set	Shall not	
Show	Shall	See 6.4.2.
Start	Shall not	
Stop	Shall not	

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

418 **6.4.1 Ordering of Results**

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

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

422 **6.4.2 Show**

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

425 **6.4.2.1 Show Command Form for a Single Instance Target – CIM_ComputerSystem Reference**

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

429 **6.4.2.1.1 Command Form**

430 `show <CIM_SystemDevice single instance>`

431 **6.4.2.1.2 CIM Requirements**

432 See `CIM_SystemDevice` in the “CIM Elements” section of the [System Memory Profile](#) for the list of
 433 mandatory properties and CIM classes that can be referenced.

434 **6.4.2.1.3 Behavior Requirements**

435 **6.4.2.1.3.1 Preconditions**

436 In this section `$instance` represents the instance of a `CIM_ComputerSystem`, which is referenced by
 437 `CIM_SystemDevice`.

438 6.4.2.1.3.2 Pseudo Code

```
439 &smShowAssociationInstances ( "CIM_SystemDevice", $instance.getObjectPath() );  
440 &smEnd;
```

441 6.4.2.2 Show Command Form for a Single Instance – CIM_Memory Reference

442 This command form is used to show a single instance of CIM_SystemDevice. This command form
443 corresponds to a `show` command issued against a single instance of CIM_SystemDevice, where only one
444 reference is specified and the reference is to the instance of CIM_Memory.

445 6.4.2.2.1 Command Form

```
446 show <CIM_SystemDevice single instance>
```

447 6.4.2.2.2 CIM Requirements

448 See CIM_SystemDevice in the “CIM Elements” section of the [System Memory Profile](#) for the list of
449 mandatory properties and CIM classes that can be referenced.

450 6.4.2.2.3 Behavior Requirements**451 6.4.2.2.3.1 Preconditions**

452 In this section `$instance` represents the instance of CIM_Memory which is referenced by
453 CIM_SystemDevice.

454 6.4.2.2.3.2 Pseudo Code

```
455 &smShowAssociationInstances ( "CIM_SystemDevice", $instance.getObjectPath() );  
456 &smEnd;
```

457 6.4.2.3 Show Command Form for a Single Instance – Both References

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

461 6.4.2.3.1 Command Form

```
462 show <CIM_SystemDevice single instance>
```

463 6.4.2.3.2 CIM Requirements

464 See CIM_SystemDevice in the “CIM Elements” section of the [System Memory Profile](#) for the list of
465 mandatory properties and CIM classes that can be referenced.

466 6.4.2.3.3 Behavior Requirements**467 6.4.2.3.3.1 Preconditions**

468 In this section `$instanceA` represents the referenced instance of CIM_Memory through
469 CIM_SystemDevice association. `$instanceB` represents the instance of CIM_Memory which is
470 referenced by CIM_SystemDevice.

471 **6.4.2.3.3.2 Pseudo Code**

```
472 &smShowAssociationInstance ( "CIM_SystemDevice", $instanceA.getObjectPath(),  
473     $instanceB.getObjectPath() );  
474 &smEnd;
```

475

476
477
478
479
480

ANNEX A
(informative)

Change Log

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

481