



1
2
3
4

Document Number: DSP0801

Date: 2009-07-29

Version: 1.0.0

5
6

Command Line Protocol Service Profile SM CLP Command 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27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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

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_BindsTo 9

45 6.2 CIM_CLPCapabilities..... 12

46 6.3 CIM_CLPProtocolEndpoint..... 13

47 6.4 CIM_CLPSettingData..... 17

48 6.5 CIM_ConcreteJob 20

49 6.6 CIM_ElementCapabilities 24

50 6.7 CIM_ElementSettingData 26

51 6.8 CIM_Error 31

52 6.9 CIM_HostedAccessPoint..... 31

53 6.10 CIM_HostedJobDestination 33

54 6.11 CIM_HostedService 35

55 6.12 CIM_JobDestinationJobs 38

56 6.13 CIM_JobQueue..... 40

57 6.14 CIM_OwningJobElement 42

58 6.15 CIM_ProtocolService 44

59 6.16 CIM_ProvidesEndpoint 49

60 6.17 CIM_ServiceAffectsElement 51

61 ANNEX A (informative) Change Log 54

62

63 Tables

64 Table 1 – Command Verb Requirements for CIM_BindsTo 10

65 Table 2 – Command Verb Requirements for CIM_CLPCapabilities..... 12

66 Table 3 – Command Verb Requirements for CIM_CLPProtocolEndpoint..... 13

67 Table 4 – Command Verb Requirements for CIM_CLPSettingData..... 17

68 Table 5 – Command Verb Requirements for CIM_ConcreteJob 20

69 Table 6 – Command Verb Requirements for CIM_ElementCapabilities 24

70 Table 7 – Command Verb Requirements for CIM_ElementSettingData 26

71 Table 8 – Command Verb Requirements for CIM_HostedAccessPoint 31

72 Table 9 – Command Verb Requirements for CIM_HostedJobDestination 33

73 Table 10 – Command Verb Requirements for CIM_HostedService 36

74 Table 11 – Command Verb Requirements for CIM_JobDestinationJobs 38

75 Table 12 – Command Verb Requirements for CIM_JobQueue..... 40

76 Table 13 – Command Verb Requirements for CIM_OwningJobElement 42

77 Table 14 – Command Verb Requirements for CIM_ProtocolService 44

78 Table 15 – Command Verb Requirements for CIM_ProvidesEndpoint 49
79 Table 16 – Command Verb Requirements for CIM_ServiceAffectsElement 51
80

81

Foreword

82 The *Command Line Protocol Service Profile SM CLP Command Mapping Specification* (DSP0801) was
83 prepared by the Server Management Working Group and the Physical Profiles Working Group of the
84 DMTF.

85 Conventions

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

88 Acknowledgements

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

- 91 • Christina Shaw – HP
- 92 • Aaron Merkin – IBM
- 93 • Jon Hass – Dell
- 94 • Khachatur Papanyan – Dell
- 95 • Jeff Hilland – HP
- 96 • Perry Vincent – Intel
- 97 • John Leung – Intel

98

99

Introduction

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

104 The target audience for this specification is implementers of the SM CLP support for the [CLP Service](#)
105 [Profile](#).

106 **Command Line Protocol Service Profile SM CLP Command** 107 **Mapping Specification**

108 **1 Scope**

109 This specification contains the requirements for an implementation of the SM CLP to provide access to,
110 and implement the behaviors of, the [CLP Service Profile](#).

111 **2 Normative References**

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

115 **2.1 Approved References**

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

118 DMTF DSP1005, *CLP Service Profile 1.0*,
119 http://www.dmtf.org/standards/published_documents/DSP1005_1.0.pdf

120 SNIA, *Storage Management Initiative Specification (SMI-S) 1.1.0*, November 2005,
121 http://www.snia.org/tech_activities/standards/curr_standards/smi/

122 **2.2 Other References**

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

125 **3 Terms and Definitions**

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

127 **3.1**

128 **can**

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

130 **3.2**

131 **cannot**

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

133 **3.3**

134 **conditional**

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

- 137 **3.4**
138 **mandatory**
139 indicates requirements to be followed strictly in order to conform to the document and from which no
140 deviation is permitted
- 141 **3.5**
142 **may**
143 indicates a course of action permissible within the limits of the document
- 144 **3.6**
145 **need not**
146 indicates a course of action permissible within the limits of the document
- 147 **3.7**
148 **optional**
149 indicates a course of action permissible within the limits of the document
- 150 **3.8**
151 **shall**
152 indicates requirements to be followed strictly in order to conform to the document and from which no
153 deviation is permitted
- 154 **3.9**
155 **shall not**
156 indicates requirements to be followed strictly in order to conform to the document and from which no
157 deviation is permitted
- 158 **3.10**
159 **should**
160 indicates that among several possibilities, one is recommended as particularly suitable, without
161 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 162 **3.11**
163 **should not**
164 indicates that a certain possibility or course of action is deprecated but not prohibited

165 **4 Symbols and Abbreviated Terms**

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

- 167 **4.1**
168 **CIM**
169 Common Information Model
- 170 **4.2**
171 **CLP**
172 Command Line Protocol
- 173 **4.3**
174 **DMTF**
175 Distributed Management Task Force

- 176 **4.4**
177 **IETF**
178 Internet Engineering Task Force
- 179 **4.5**
180 **SM**
181 Server Management
- 182 **4.6**
183 **SMI-S**
184 Storage Management Initiative Specification
- 185 **4.7**
186 **SNIA**
187 Storage Networking Industry Association
- 188 **4.8**
189 **UFsT**
190 User Friendly selection Tag

191 **5 Recipes**

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

- 194 • smStartRSC()
- 195 • smStopRSC()
- 196 • smResetRSC()
- 197 • smShowInstance()
- 198 • smShowInstances()
- 199 • smSetInstance()
- 200 • smShowAssociationInstances()
- 201 • smShowAssociationInstance()
- 202 • smDeleteInstance
- 203 • smMakeCommandStatus
- 204 • smNewInstance

205 **6 Mappings**

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

209 **6.1 CIM_BindsTo**

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

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

216 **Table 1 – Command Verb Requirements for CIM_BindsTo**

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	

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

219 **6.1.1 Ordering of Results**

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

- 222 • Results for CIM_BindsTo are unordered; therefore, no algorithm is defined.

223 **6.1.2 Show**

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

226 The `show` command is used to display information about the CIM_BindsTo instance or instances.

227 **6.1.2.1 Show a Single Instances – CIM_SSHProtocolEndpoint or CIM_TelnetProtocolEndpoint**

228 This command form is for the `show` verb applied to multiple instances. This command form corresponds
 229 to the `show` command issued against CIM_BindsTo where only one reference is specified and the
 230 reference is to an instance of CIM_SSHProtocolEndpoint or CIM_TelnetProtocolEndpoint.

231 **6.1.2.1.1 Command Form**

232 `show <CIM_BindsTo multiple instances>`

233 **6.1.2.1.2 CIM Requirements**

234 See CIM_BondsTo in the “CIM Elements” section of the [CLP Service Profile](#) for the list of mandatory
 235 properties.

236 6.1.2.1.3 Behavior Requirements

237 6.1.2.1.3.1 Preconditions

238 \$instance contains the instance of CIM_SSHProtocolEndpoint or
239 CIM_TelnetProtocolEndpoint which is referenced by CIM_BindsTo

240 6.1.2.1.3.2 Pseudo Code

```
241 &smShowAssociationInstances ( "CIM_BindsTo", $instance.getObjectPath() );
242 smEnd;
```

243 6.1.2.2 Show a Single Instance – CIM_CLPProtocolEndpoint Reference

244 This command form is for the `show` verb applied to a single instance. This command form corresponds to
245 the `show` command issued against `CIM_BindsTo` where the reference specified is to an instance of
246 `CIM_CLPProtocolEndpoint`. A single instance of `CIM_SSHProtocolEndpoint` or
247 `CIM_TelnetProtocolEndpoint` is associated with each instance of `CIM_CLPProtocolEndpoint`. Therefore, a
248 single instance will be returned.

249 6.1.2.2.1 Command Form

```
250 show <CIM_BindsTo single instance>
```

251 6.1.2.2.2 CIM Requirements

252 See the “CIM Elements” section of the [CLP Service Profile](#) for the list of mandatory properties.

253 6.1.2.2.3 Behavior Requirements

254 6.1.2.2.3.1 Preconditions

255 \$instance contains the instance of CIM_CLPProtocolEndpoint which is referenced by
256 CIM_BindsTo

257 6.1.2.2.3.2 Pseudo Code

```
258 &smShowAssociationInstances ( "CIM_BindsTo", $instance.getObjectPath() );
259 smEnd;
```

260 6.1.2.3 Show a Single Instance – Both References

261 This command form is for the `show` verb applied to a single instance. This command form corresponds to
262 the `show` command issued against `CIM_BindsTo` where a reference to `CIM_CLPProtocolEndpoint` and a
263 reference to `CIM_SSHProtocolEndpoint` or `CIM_TelnetProtocolEndpoint` are specified and therefore the
264 desired instance is unambiguously identified.

265 6.1.2.3.1 Command Form

```
266 show <CIM_BindsTo single instance>
```

267 6.1.2.3.2 CIM Requirements

268 See the “CIM Elements” section of the [CLP Service Profile](#) for the list of mandatory properties.

269 6.1.2.3.3 Behavior Requirements

270 6.1.2.3.3.1 Preconditions

271 1) \$instanceA contains the instance of CIM_CLPProtocolEndpoint which is referenced by

272 CIM_BindsTo
 273 2) \$instanceB contains the instance of CIM_ProtocolEndpoint which is referenced by
 274 CIM_BindsTo.

275 6.1.2.3.3.2 Pseudo Code

```
276 &smShowAssociationInstance ( "CIM_BindsTo",
277     $instanceA.getObjectPath(), $instanceB.getObjectPath() );
278 smEnd;
```

279 6.2 CIM_CLPCapabilities

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

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

286 **Table 2 – Command Verb Requirements for CIM_CLPCapabilities**

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	

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

289 6.2.1 Ordering of Results

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

- 292 • Results for CIM_CLPCapabilities are unordered; therefore, no algorithm is defined.

293 6.2.2 Show

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

296 The `show` verb is used to display information about an instance or instances of the CIM_CLPCapabilities
 297 class.

298 6.2.2.1 Show a Single Instance

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

300 6.2.2.1.1 Command Form

```
301 show <CIM_CLPCapabilities single instance>
```

302 6.2.2.1.2 CIM Requirements

303 See `CIM_CLPCapabilities` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
304 mandatory properties.

305 6.2.2.1.3 Behavior Requirements

306 6.2.2.1.3.1 Preconditions

```
307 #all is true if the all option was specified with the command; otherwise, #all is  
308 false
```

309 6.2.2.1.3.2 Pseudo Code

```
310 $instance=<CIM_CLPCapabilities Single Instance>  
311 #propertylist[] = null;  
312 if ( false == #all) {  
313     #propertylist[] = { //all mandatory non-key properties }  
314 }  
315 &smShowInstance($instance.getObjectPath(), #propertylist[]);  
316 &smEnd;
```

317 6.3 CIM_CLPProtocolEndpoint

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

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

324 **Table 3 – Command Verb Requirements for CIM_CLPProtocolEndpoint**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.3.2.
show	Shall	See 6.3.3.
start	Not supported	
stop	May	See 6.3.4.

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

327 **6.3.1 Ordering of Results**

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

- 330 • Results for `CIM_CLPProtocolEndpoint` are unordered; therefore, no algorithm is defined.

331 **6.3.2 Set**

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

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

336 **6.3.2.1 General Usage of Set for a Single Property**

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

339 The requirement for supporting modification of a property using this command form shall be equivalent to
340 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined
341 in the [CLP Service Profile](#).

342 **6.3.2.1.1 Command Form**

```
343 set <CIM_CLPProtocolEndpoint single instance> <propertyname>=<propertyvalue>
```

344 **6.3.2.1.2 CIM Requirements**

345 See `CIM_CLPProtocolEndpoint` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
346 modifiable properties.

347 **6.3.2.1.3 Behavior Requirements**

```
348 $instance = <CIM_CLPProtocolEndpoint Single Instance>
349 #propertyName[] = {<propertyname>};
350 #propertyValues[] = {<propertyvalue>};
351 &smSetInstance($instance, #propertyName[], #propertyValues[]);
352 &smEnd;
```

353 **6.3.2.2 General Usage of Set for Multiple Properties**

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

357 The requirement for supporting modification of a property using this command form shall be equivalent to
358 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined
359 in the [CLP Service Profile](#).

360 **6.3.2.2.1 Command Form**

```
361 set <CIM_CLPProtocolEndpoint single instance> <propertyname1>=<propertyvalue1>
362 <propertynamen>=<propertyvaluen>
```

363 6.3.2.2.2 CIM Requirements

364 See CIM_CLPProtocolEndpoint in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
365 mandatory properties.

366 6.3.2.2.3 Behavior Requirements

```
367 $instance = <CIM_CLPProtocolEndpoint Single Instance>
368 #propertyName[] = {<propertyname>};
369
370 for #i < n
371 {
372     #propertyName[#i] = <propertyname#i>
373     #propertyValue[#i] = <propertyvalue#i>
374 }
375
376 &smSetInstance($instance, #propertyName[], #propertyValue[]);
377 &smEnd;
```

378 6.3.3 Show

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

382 The `show` verb is used to display information about a CLP session.

383 6.3.3.1 Show a Single Instance

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

385 6.3.3.1.1 Command Form

```
386 show <CIM_CLPProtocolEndpoint single instance>
```

387 6.3.3.1.2 CIM Requirements

388 See CIM_CLPProtocolEndpoint in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
389 mandatory properties.

390 6.3.3.1.3 Behavior Requirements

391 6.3.3.1.3.1 Preconditions

```
392 #all is true if the all option was specified with the command; otherwise, #all is
393 false
```

394 6.3.3.1.3.2 Pseudo Code

```
395 $instance=<CIM_CLPProtocolEndpoint Single Instance>
396 #propertylist[] = null;
397 if ( false == #all) {
398     #propertylist[] = { //all mandatory non-key properties }
399 }
400 &smShowInstance($instance.getObjectPath(), #propertylist[]);
```

401 6.3.3.2 Show Multiple Instances Scoped by System

402 This command form is for the `show` verb applied to multiple instances of `CIM_CLPProtocolEndpoint`. This
403 command form corresponds to UFsT-based selection within a scoping system.

404 6.3.3.2.1 Command Form

```
405 show <CIM_CLPProtocolEndpoint multiple instances>
```

406 6.3.3.2.2 CIM Requirements

407 See `CIM_CLPProtocolEndpoint` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
408 mandatory properties.

409 6.3.3.2.3 Behavior Requirements

410 6.3.3.2.3.1 Preconditions

```
411 1) $containerInstance contains the instance of CIM_ComputerSystem for which we are
412 displaying scoped endpoints (CIM_CLPProtocolEndpoint instances). The CLP Service
413 Profile requires that the CIM_CLPProtocolEndpoint instance be associated with its
414 scoping system via an instance of the CIM_HostedAccessPoint association.
415 2) #all is true if the all option was specified with the command; otherwise, #all is
416 false
```

417 6.3.3.2.3.2 Pseudo Code

```
418 #propertylist[] = null;
419 if (false == #all) {
420     #propertylist[] = { //all mandatory non-key properties }
421 }
422
423 &smShowInstances ( "CIM_CLPProtocolEndpoint", "CIM_HostedAccessPoint",
424 $containerInstance.getObjectPath(), #propertylist[] );
425 &smEnd;
```

426 6.3.4 Stop

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

430 The `stop` verb is used to terminate a CLP session.

431 6.3.4.1 Stop a Single Instance

432 This command form is for the `stop` verb applied to a single instance of `CIM_CLPProtocolEndpoint`. The
433 lifecycle of a CLP session corresponds to the lifecycle of the `CIM_CLPProtocolEndpoint` which represents
434 it. Therefore, stopping a CLP service corresponds to a deletion of the underlying instance.

435 6.3.4.1.1 Command Form

```
436 stop <CIM_CLPProtocolEndpoint single instance>
```

437 6.3.4.1.2 CIM Requirements

438 See `CIM_CLPProtocolEndpoint` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
439 mandatory properties.

440 6.3.4.1.3 Behavior Requirements

```
441 $instance=<CIM_CLPProtocolEndpoint Single Instance>
442 &smDeleteInstance($instance.GetObjectPath());
443 &smEnd;
```

444 6.4 CIM_CLPSettingData

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

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

451 **Table 4 – Command Verb Requirements for CIM_CLPSettingData**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.4.2.
show	Shall	See 6.4.3.
start	Not supported	
stop	Not supported	

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

453 6.4.1 Ordering of Results

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

- 456 • Results for `CIM_CLPSettingData` are unordered; therefore, no algorithm is defined.

457 6.4.2 Set

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

460 The `set` verb is used to modify configuration represented by an instance of `CIM_CLPSettingData`.

461 6.4.2.1 General Usage of Set for a Single Property

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

464 The requirement for supporting modification of a property using this command form shall be equivalent to
 465 the requirement for supporting modification of the property using the ModifyInstance operation as defined
 466 in the [CLP Service Profile](#).

467 6.4.2.1.1 Command Form

```
468 set <CIM_CLPSettingData single instance> <propertyname>=<propertyvalue>
```

469 6.4.2.1.2 CIM Requirements

470 See CIM_CLPSettingData in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
 471 modifiable properties.

472 6.4.2.1.3 Behavior Requirements

```
473 $instance = <CIM_CLPSettingData Single Instance>
474 #propertyNames[] = {<propertyname>};
475 #propertyValues[] = {<propertyvalue>};
476 &smSetInstance($instance, #propertyNames[], #propertyValues[]);
477 &smEnd;
```

478 6.4.2.2 General Usage of Set for Multiple Properties

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

482 The requirement for supporting modification of a property using this command form shall be equivalent to
 483 the requirement for supporting modification of the property using the ModifyInstance operation as defined
 484 in the [CLP Service Profile](#).

485 6.4.2.2.1 Command Form

```
486 set <CIM_CLPSettingData single instance> <propertyname1>=<propertyvalue1>
487 <propertynamen>=<propertyvaluen>
```

488 6.4.2.2.2 CIM Requirements

489 See CIM_CLPSettingData in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
 490 mandatory properties.

491 6.4.2.2.3 Behavior Requirements

```
492 $instance = <CIM_CLPSettingData Single Instance>
493 #propertyNames[] = {<propertyname>};
494 for #i < n
495 {
496     #propertyNames[#i] = <propertyname#i>
497     #propertyValues[#i] = <propertyvalue#i>
498 }
499 &smSetInstance($instance, #propertyNames[], #propertyValues[]);
500 &smEnd;
```

501 **6.4.3 Show**

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

505 The `show` verb is used to display information about the `CIM_CLPSettingData` instance.

506 **6.4.3.1 Show a Single Instance**

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

508 **6.4.3.1.1 Command Form**

```
509 show <CIM_CLPSettingData single instance>
```

510 **6.4.3.1.2 CIM Requirements**

511 See `CIM_CLPSettingData` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
512 mandatory properties.

513 **6.4.3.1.3 Behavior Requirements**

514 **6.4.3.1.3.1 Preconditions**

```
515 #all is true if the all option was specified with the command; otherwise, #all is  
516 false
```

517 **6.4.3.1.3.2 Pseudo Code**

```
518 $instance=<CIM_CLPSettingData Single Instance>  
519 &lShowPEndpoint($instance, #all);  
520 &smEnd;
```

521 **6.4.3.2 Show Multiple Instances Scoped by ConcreteCollection**

522 This command form is for the `show` verb applied to multiple instances of `CIM_CLPSettingData`. This
523 command form corresponds to UFsT-based selection within an instance of `CIM_ConcreteCollection`.

524 **6.4.3.2.1 Command Form**

```
525 show <CIM_CLPSettingData multiple instances>
```

526 **6.4.3.2.2 CIM Requirements**

527 See `CIM_CLPSettingData` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
528 mandatory properties.

529 **6.4.3.2.3 Behavior Requirements**

530 **6.4.3.2.3.1 Preconditions**

```
531 1) $containerInstance contains the instance of CIM_ConcreteCollection for which we are  
532 displaying contained CIM_CLPSettingData instances. The SMASH Collections Profile  
533 requires that the CIM_CLPSettingData instances be aggregated into an addressing  
534 collection via CIM_MemberOfCollection.  
535 2) #all is true if the all option was specified with the command; otherwise, #all is  
536 false
```

537 6.4.3.2.3.2 Pseudo Code

```

538 #propertylist[] = null;
539 //this property list will match the property list in lShowPEndpoint()
540 if (false == #all) {
541     #propertylist[] = { //all mandatory non-key properties }
542 }
543
544 &smShowInstances ( "CIM_CLPSettingData", "CIM_MemberOfCollection",
545 $containerInstance.getObjectPath(), #propertylist[] );
546 &smEnd;

```

547 6.5 CIM_ConcreteJob

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

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

554 **Table 5 – Command Verb Requirements for CIM_ConcreteJob**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.5.2.
show	Shall	See 6.5.3.
start	Not supported	
stop	May	See 6.5.4.

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

557 6.5.1 Ordering of Results

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

- 560 • Results for `CIM_ConcreteJob` are unordered; therefore, no algorithm is defined.

561 6.5.2 Set

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

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

565 6.5.2.1 General Usage of Set for a Single Property

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

568 The requirement for supporting modification of a property using this command form shall be equivalent to
569 the requirement for supporting modification of the property using the ModifyInstance operation as defined
570 in the [CLP Service Profile](#).

571 6.5.2.1.1 Command Form

```
572 set <CIM_ConcreteJob single instance> <propertyname>=<propertyvalue>
```

573 6.5.2.1.2 CIM Requirements

574 See CIM_ConcreteJob in the “CIM Elements” section of the [CLP Service Profile](#) for the list of modifiable
575 properties.

576 6.5.2.1.3 Behavior Requirements

```
577 $instance = <CIM_ConcreteJob Single Instance>
578 #propertyName[] = {<propertyname>};
579 #propertyValues[] = {<propertyvalue>};
580 &smSetInstance($instance, #propertyName[], #propertyValues[]);
581 &smEnd;
```

582 6.5.2.2 General Usage of Set for Multiple Properties

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

586 The requirement for supporting modification of a property using this command form shall be equivalent to
587 the requirement for supporting modification of the property using the ModifyInstance operation as defined
588 in the [CLP Service Profile](#).

589 6.5.2.2.1 Command Form

```
590 set <CIM_ConcreteJob single instance> <propertyname1>=<propertyvalue1>
591 <propertynamen>=<propertyvaluen>
```

592 6.5.2.2.2 CIM Requirements

593 See CIM_ConcreteJob in the “CIM Elements” section of the [CLP Service Profile](#) for the list of mandatory
594 properties.

595 6.5.2.2.3 Behavior Requirements

```
596 $instance = <CIM_ConcreteJob Single Instance>
597 #propertyName[] = {<propertyname>};
598
599 for #i < n
600 {
601     #propertyName[#i] = <propertyname#i>
602     #propertyValues[#i] = <propertyvalue#i>
603 }
604
```

```
605 &smSetInstance($instance, #propertyName[], #propertyValues[]);  
606 &smEnd;
```

607 **6.5.3 Show**

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

610 The `show` verb is used to display information about the CLP Service's Job Queue.

611 **6.5.3.1 Show a Single Instance**

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

613 **6.5.3.1.1 Command Form**

```
614 show <CIM_ConcreteJob single instance>
```

615 **6.5.3.1.2 CIM Requirements**

616 See `CIM_ConcreteJob` in the "CIM Elements" section of the [CLP Service Profile](#) for the list of mandatory
617 properties.

618 **6.5.3.1.3 Behavior Requirements**

619 **6.5.3.1.3.1 Preconditions**

```
620 #all is true if the all option was specified with the command; otherwise, #all is  
621 false
```

622 **6.5.3.1.3.2 Pseudo Code**

```
623 $instance=<CIM_ConcreteJob Single Instance>  
624 #propertylist[] = null;  
625 if ( false == #all) {  
626     #propertylist[] = { //all mandatory non-key properties }  
627 }  
628 &smShowInstance($instance.getObjectPath(), #propertylist[]);
```

629 **6.5.3.2 Show Multiple Instances Scoped by JobQueue**

630 This command form is for the `show` verb applied to multiple instances of `CIM_ConcreteJob`. This
631 command form corresponds to UFT-based selection within a scoping Job Queue.

632 **6.5.3.2.1 Command Form**

```
633 show <CIM_ConcreteJob multiple instances>
```

634 **6.5.3.2.2 CIM Requirements**

635 See `CIM_ConcreteJob` in the "CIM Elements" section of the [CLP Service Profile](#) for the list of mandatory
636 properties.

637 **6.5.3.2.3 Behavior Requirements**

638 **6.5.3.2.3.1 Preconditions**

```
639 1) $containerInstance contains the instance of CIM_JobQueue for which we are  
640 displaying scoped jobs (CIM_ConcreteJob instances). The CLP Service Profile
```

```

641     requires that the CIM_ConcreteJob instance be associated with its scoping job queue
642     via an instance of the CIM_JobDestinationJobs association.
643 2) #all is true if the all option was specified with the command; otherwise, #all is
644     false

```

645 6.5.3.2.3.2 Pseudo Code

```

646 #propertylist[] = null;
647 if (false == #all) {
648     #propertylist[] = { //all mandatory non-key properties }
649 }
650
651 &smShowInstances ( "CIM_ConcreteJob", "CIM_JobDestinationJobs",
652     $containerInstance.getObjectPath(), #propertylist[] );
653 &smEnd;

```

654 6.5.4 Stop

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

657 The `stop` verb is used to terminate or kill a CLP operation.

658 6.5.4.1 Stop a Single Instance

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

660 6.5.4.1.1 Command Form

```

661 stop [-f] <CIM_ProtocolService single instance>

```

662 6.5.4.1.2 CIM Requirements

```

663 uint16 EnabledState;
664 uint16 RequestedState;
665 uint32 EnabledLogicalElement.RequestStateChange (
666     [IN] uint16 RequestedState,
667     [OUT] REF CIM_ConcreteJob Job,
668     [IN] datetime TimeoutPeriod );

```

669 6.5.4.1.3 Behavior Requirements

670 6.5.4.1.3.1 Preconditions

```

671 #force is true if force option was specified; otherwise, #force is false
672 if (#force) {
673     #requestedState = 5;//kill
674 }
675 else {
676     #requestedstate = 4;//terminate
677 }
678 $instance=<CIM_ProtocolService Single Instance>
679 &smRequestStateChange ( $instance.getObjectPath(), #requestedState );
680 smEnd;

```

681 **6.6 CIM_ElementCapabilities**

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

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

688 **Table 6 – 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 6.6.2.
start	Not supported	
stop	Not supported	

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

691 **6.6.1 Ordering of Results**

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

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

695 **6.6.2 Show**

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

699 The `show` command is used to display information about the `CIM_ElementCapabilities` instance or
 700 instances.

701 **6.6.2.1 Show Multiple Instances – CIM_CLPCapabilities Reference**

702 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 703 the `show` command issued against `CIM_ElementCapabilities` where the reference specified is to an
 704 instance of `CIM_CLPCapabilities`. Multiple instances of `CIM_ProtocolService` can be associated with each
 705 instance of a `CIM_CLPCapabilities`.

706 **6.6.2.1.1 Command Form**

707 `show <CIM_ElementCapabilities multiple instances>`

708 6.6.2.1.2 CIM Requirements

709 See CIM_ElementCapabilities in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
710 mandatory properties.

711 6.6.2.1.3 Behavior Requirements

712 6.6.2.1.3.1 Preconditions

713 \$instance contains the instance of CIM_CLPCapabilities which is referenced by
714 CIM_ElementCapabilities

715 6.6.2.1.3.2 Pseudo Code

```
716 &smShowAssociationInstances ( "CIM_ElementCapabilities", $instance.getObjectPath() );  
717     smEnd;
```

718 6.6.2.2 Show a Single Instance – CIM_ProtocolService Reference

719 This command form is for the `show` verb applied to a single instance. This command form corresponds to
720 the `show` command issued against CIM_ElementCapabilities where the reference specified is to an
721 instance of CIM_ProtocolService. A single instance of CIM_CLPCapabilities is associated with each
722 instance of CIM_ProtocolService. Therefore, a single instance will be returned.

723 6.6.2.2.1 Command Form

```
724 show <CIM_ElementCapabilities single instance>
```

725 6.6.2.2.2 CIM Requirements

726 See CIM_ElementCapabilities in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
727 mandatory properties.

728 6.6.2.2.3 Behavior Requirements

729 6.6.2.2.3.1 Preconditions

730 \$instance contains the instance of CIM_ProtocolService which is referenced by
731 CIM_ElementCapabilities

732 6.6.2.2.3.2 Pseudo Code

```
733 &smShowAssociationInstances ( "CIM_ElementCapabilities", $instance.getObjectPath() );  
734     smEnd;
```

735 6.6.2.3 Show a Single Instance – Both References

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

739 6.6.2.3.1 Command Form

```
740 show <CIM_ElementCapabilities single instance>
```

741 6.6.2.3.2 CIM Requirements

742 See CIM_ElementCapabilities in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
743 mandatory properties.

744 6.6.2.3.3 Behavior Requirements

745 6.6.2.3.3.1 Preconditions

- 746 1) \$instanceA contains the instance of CIM_CLPCapabilities which is referenced by
 747 CIM_ElementCapabilities
 748 2) \$instanceB contains the instance of CIM_ProtocolService which is referenced by
 749 CIM_ElementCapabilities.

750 6.6.2.3.3.2 Pseudo Code

```
751 &smShowAssociationInstance ( "CIM_ElementCapabilities",
752     $instanceA.getObjectPath(),$instanceB.getObjectPath() );
753 smEnd;
```

754 6.7 CIM_ElementSettingData

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

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

761 **Table 7 – Command Verb Requirements for CIM_ElementSettingData**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.7.2.
show	Shall	See 6.7.3.
start	Not supported	
stop	Not supported	

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

764 6.7.1 Ordering of Results

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

- 767 • Results for CIM_ElementSettingData are unordered; therefore, no algorithm is defined.

768 6.7.2 Set

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

772 The `set` verb is used to modify properties of the `CIM_ElementSettingData` instance.

773 6.7.2.1 Set of `IsNext`

774 The `IsNext` property is the only property of `CIM_ElementSettingData` which can be modified directly via
775 the `set` verb.

776 6.7.2.1.1 Command Form

```
777 set <CIM_ElementSettingData single instance> IsNext=<propertyvalue>
```

778 6.7.2.1.2 CIM Requirements

779 See `CIM_ElementSettingData` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
780 mandatory properties.

781 6.7.2.1.3 Behavior Requirements

```
782 $instance = <CIM_ElementSettingData Single Instance>
783 #propertyName[] = {"IsNext"};
784 #propertyValues[] = {<propertyvalue>};
785 &smSetInstance($instance, #propertyName[], #propertyValues[]);
786 &smEnd;
```

787 6.7.3 Show

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

791 The `show` command is used to display information about the `CIM_ElementSettingData` instance or
792 instances.

793 6.7.3.1 Show Multiple Instances – `CIM_CLPSettingData` and `CIM_CLPProtocolEndpoint`

794 This command form corresponds to the `show` command issued against `CIM_ElementSettingData` where
795 the reference specified is to an instance of `CIM_CLPSettingData`. Note that when an instance of
796 `CIM_CLPSettingData` is associated with an instance of `CIM_CLPProtocolEndpoint`, the `IsCurrent` property
797 is the mandatory property.

798 6.7.3.1.1 Command Form

```
799 show <CIM_ElementSettingData multiple instances>
```

800 6.7.3.1.2 CIM Requirements

801 See `CIM_ElementSettingData` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
802 mandatory properties.

803 6.7.3.1.3 Behavior Requirements

804 6.7.3.1.3.1 Preconditions

- ```
805 1) $instance contains the instance of CIM_CLPSettingData which is referenced by
806 CIM_ElementSettingData
807 2) #all is true if the all option was specified
```

### 808 6.7.3.1.3.2 Pseudo Code

```

809 #propertylist = NULL;
810 if (false == #all) {
811 #propertylist = { "IsCurrent" };
812 }
813 &smShowAssociationInstances ("CIM_ElementSettingData", $instance.getObjectPath(),
814 #propertylist[]);
815 smEnd;

```

### 816 6.7.3.2 Show Multiple Instances – CIM\_CLPProtocolEndpoint Reference

817 This command form corresponds to the `show` command issued against `CIM_ElementSettingData` where  
 818 the reference specified is to an instance of `CIM_CLPProtocolEndpoint`. Note that when an instance of  
 819 `CIM_CLPSettingData` is associated with an instance of `CIM_CLPProtocolEndpoint`, the `IsCurrent` property  
 820 is the mandatory property.

#### 821 6.7.3.2.1 Command Form

```
822 show <CIM_ElementSettingData multiple instances>
```

#### 823 6.7.3.2.2 CIM Requirements

824 See `CIM_ElementSettingData` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
 825 mandatory properties.

#### 826 6.7.3.2.3 Behavior Requirements

##### 827 6.7.3.2.3.1 Preconditions

- ```

828 1) $instance contains the instance of CIM_CLPProtocolEndpoint which is referenced by
829     CIM_ElementSettingData
830 2) #all is true if the all option was specified
831

```

832 6.7.3.2.3.2 Pseude Code

```

833 #propertylist = NULL;
834 if (false == #all) {
835     #propertylist = { "IsCurrent" };
836 }
837 &smShowAssociationInstances ( "CIM_ElementSettingData", $instance.getObjectPath(),
838     #propertylist[] );
839 smEnd;

```

840 6.7.3.3 Show a Single Instance – CIM_CLPSettingData and CIM_CLPProtocolEndpoint

841 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 842 the `show` command issued against `CIM_ElementSettingData` where both references are specified and
 843 therefore the desired instance is unambiguously identified.

844 6.7.3.3.1 Command Form

```
845 show <CIM_ElementSettingData single instance>
```

846 6.7.3.3.2 CIM Requirements

847 See CIM_ElementSettingData in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
848 mandatory properties.

849 6.7.3.3.3 Behavior Requirements

850 6.7.3.3.3.1 Preconditions

- 851 1) \$instanceA contains the instance of CIM_CLPSettingData which is referenced by
852 CIM_ElementSettingData
- 853 2) \$instanceB contains the instance of CIM_CLPProtocolEndpoint which is referenced by
854 CIM_ElementSettingData.
- 855 3) #all is true if the all option was specified

856 6.7.3.3.3.2 Pseudo Code

```
857 #propertylist = NULL;
858 if (false == #all) {
859     #propertylist = { "IsCurrent" };
860 }
861 &smShowAssociationInstance ( "CIM_ElementSettingData",
862     $instanceA.getObjectPath(), $instanceB.getObjectPath(),
863     #propertylist[] );
864 smEnd;
```

865 6.7.3.4 Show Multiple Instances – CIM_CLPSettingData and CIM_ProtocolService

866 This command form corresponds to the `show` command issued against CIM_ElementSettingData where
867 the reference specified is to an instance of CIM_CLPSettingData. Note that when an instance of
868 CIM_CLPSettingData is associated with an instance of CIM_ProtocolService, the `IsNext` and `IsDefault`
869 properties are mandatory.

870 6.7.3.4.1 Command Form

```
871 show <CIM_ElementSettingData multiple instances>
```

872 6.7.3.4.2 CIM Requirements

873 See CIM_ElementSettingData in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
874 mandatory properties.

875 6.7.3.4.3 Behavior Requirements

876 6.7.3.4.3.1 Preconditions

- 877 1) \$instance contains the instance of CIM_CLPSettingData which is referenced by
878 CIM_ElementSettingData
- 879 2) #all is true if the all option was specified

880 6.7.3.4.3.2 Pseudo Code

```
881 #propertylist[] = NULL;
882 if (false == #all) {
883     #propertylist = { "IsNext", "IsDefault" };
884 }
885 &smShowAssociationInstances ( "CIM_ElementSettingData", $instance.getObjectPath(),
```

```
886 #propertylist[] );
887 smEnd;
```

888 6.7.3.5 Show Multiple Instances – CIM_ProtocolService Reference

889 This command form corresponds to the `show` command issued against `CIM_ElementSettingData` where
890 the reference specified is to an instance of `CIM_ProtocolService`. Note that when an instance of
891 `CIM_CLPSettingData` is associated with an instance of `CIM_ProtocolService`, the `IsNext` and `IsDefault`
892 properties are mandatory.

893 6.7.3.5.1 Command Form

```
894 show <CIM_ElementSettingData multiple instances>
```

895 6.7.3.5.2 CIM Requirements

896 See `CIM_ElementSettingData` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
897 mandatory properties.

898 6.7.3.5.3 Behavior Requirements

899 6.7.3.5.3.1 Preconditions

- ```
900 1) $instance contains the instance of CIM_ProtocolService which is referenced by
901 CIM_ElementSettingData
902 2) #all is true if the all option was specified
```

##### 903 6.7.3.5.3.2 Pseudo Code

```
904 #propertylist[] = NULL;
905 if (false == #all) {
906 #propertylist = { "IsNext", "IsDefault" };
907 }
908 &smShowAssociationInstances ("CIM_ElementSettingData", $instance.getObjectPath(),
909 #propertylist[]);
910 smEnd;
```

### 911 6.7.3.6 Show a Single Instance – CIM\_CLPSettingData and CIM\_ProtocolService

912 This command form is for the `show` verb applied to a single instance. This command form corresponds to  
913 the `show` command issued against `CIM_ElementSettingData` where both references are specified and  
914 therefore the desired instance is unambiguously identified.

#### 915 6.7.3.6.1 Command Form

```
916 show <CIM_ElementSettingData single instance>
```

#### 917 6.7.3.6.2 CIM Requirements

918 See `CIM_ElementSettingData` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
919 mandatory properties.

#### 920 6.7.3.6.3 Behavior Requirements

##### 921 6.7.3.6.3.1 Preconditions

- ```
922 1) $instanceA contains the instance of CIM_CLPSettingData which is referenced by
923    CIM_ElementSettingData
```

- 924 2) \$instanceB contains the instance of CIM_ProtocolService which is referenced by
- 925 CIM_ElementSettingData.
- 926 3) #all is true if the all option was specified

927 **6.7.3.6.3.2 Pseudo Code**

```

928 #propertylist[] = NULL;
929 if (false == #all) {
930     #propertylist = { "IsNext", "IsDefault" };
931 }
932 &smShowAssociationInstance ( "CIM_ElementSettingData",
933     $instanceA.getObjectPath(), $instanceB.getObjectPath(), #propertylist[] );
934 smEnd;
    
```

935 **6.8 CIM_Error**

936 CIM_Error is utilized as a template for embedded instances. Instances of the class are not exposed by
 937 the underlying profile. Therefore, the usage of any CLP verbs shall not be supported for instances of
 938 CIM_Error.

939 **6.9 CIM_HostedAccessPoint**

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

941 Table 8 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 942 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 943 verb and target. Table 8 is for informational purposes only; in case of a conflict between Table 8 and
 944 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 945 information in Table 8.

946 **Table 8 – Command Verb Requirements for CIM_HostedAccessPoint**

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 6.9.2.
start	Not supported	
stop	Not supported	

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

949 **6.9.1 Ordering of Results**

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

- 952 • Results for CIM_HostedAccessPoint are unordered; therefore, no algorithm is defined.

953 **6.9.2 Show**

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

957 The `show` command is used to display information about the `CIM_HostedAccessPoint` instance or
958 instances.

959 **6.9.2.1 Show Multiple Instances**

960 This command form is for the `show` verb applied to multiple instances. This command form corresponds
961 to a `show` command issued against `CIM_HostedAccessPoint` where only one reference is specified and
962 the reference is to an instance of `CIM_ComputerSystem`.

963 **6.9.2.2 Command Form**

```
964 show <CIM_HostedAccessPoint multiple instances>
```

965 **6.9.2.2.1 CIM Requirements**

966 See `CIM_HostedAccessPoint` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
967 mandatory properties.

968 **6.9.2.2.2 Behavior Requirements**

969 **6.9.2.2.2.1 Preconditions**

```
970 1) $instance contains the instance of CIM_ComputerSystem which is referenced by  
971 CIM_HostedAccessPoint
```

972 **6.9.2.2.2.2 Pseudo Code**

```
973 &smShowAssociationInstances ( "CIM_HostedAccessPoint", $instance.getObjectPath() );  
974 smEnd;
```

975 **6.9.2.3 Show a Single Instance – CIM_CLPProtocolEndpoint Reference**

976 This command form is for the `show` verb applied to a single instance. This command form corresponds to
977 the `show` command issued against `CIM_HostedAccessPoint` where the reference specified is to an
978 instance of `CIM_CLPProtocolEndpoint`. A single instance will be returned.

979 **6.9.2.3.1 Command Form**

```
980 show <CIM_HostedAccessPoint single instance>
```

981 **6.9.2.3.2 CIM Requirements**

982 See `CIM_HostedAccessPoint` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
983 mandatory properties.

984 **6.9.2.3.3 Behavior Requirements**

985 **6.9.2.3.3.1 Preconditions**

```
986 1) $instance contains the instance of CIM_ProtocolEndpoint that is referenced by  
987 CIM_HostedAccessPoint
```


988 **6.9.2.3.3.2 Pseudo Code**

```
989 &smShowAssociationInstances ( "CIM_HostedAccessPoint", $instance.getObjectPath() );
990 smEnd;
```

991 **6.9.2.4 Show a Single Instance – Both References**

992 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 993 the `show` command issued against `CIM_HostedAccessPoint` where both references are specified and
 994 therefore the desired instance is unambiguously identified.

995 **6.9.2.4.1 Command Form**

```
996 show <CIM_HostedAccessPoint single instance>
```

997 **6.9.2.4.2 CIM Requirements**

998 See `CIM_HostedAccessPoint` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
 999 mandatory properties.

1000 **6.9.2.4.3 Behavior Requirements**

1001 **6.9.2.4.3.1 Preconditions**

- 1002 1) `$instanceA` contains the instance of `CIM_ComputerSystem` which is referenced by
- 1003 `CIM_HostedAccessPoint`
- 1004 2) `$instanceB` contains the instance of `CIM_CLPProtocolEndpoint` that is referenced by
- 1005 `CIM_HostedAccessPoint`

1006 **6.9.2.4.3.2 Pseudo Code**

```
1007 &smShowAssociationInstance ( "CIM_HostedAccessPoint",
1008 $instanceA.getObjectPath(), $instanceB.getObjectPath() );
1009 smEnd;
```

1010 **6.10 CIM_HostedJobDestination**

1011 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

1012 Table 9 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 1013 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 1014 verb and target. Table 9 is for informational purposes only; in case of a conflict between Table 9 and
 1015 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 1016 information in Table 9.

1017 **Table 9 – Command Verb Requirements for CIM_HostedJobDestination**

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 6.10.2.

Command Verb	Requirement	Comments
start	Not supported	
stop	Not supported	

1018 No mapping is defined for the following verbs for the specified target: create, delete, dump, load,
1019 reset, set, start, and stop.

1020 6.10.1 Ordering of Results

1021 When results are returned for multiple instances of CIM_HostedJobDestination, implementations shall
1022 utilize the following algorithm to produce the natural (that is, default) ordering:

- 1023 • Results for CIM_HostedJobDestination are unordered; therefore, no algorithm is defined.

1024 6.10.2 Show

1025 This section describes how to implement the `show` verb when applied to an instance of
1026 CIM_HostedJobDestination. Implementations shall support the use of the `show` verb with
1027 CIM_HostedJobDestination.

1028 The `show` command is used to display information about the CIM_HostedJobDestination instance or
1029 instances.

1030 6.10.2.1 Show a Single Instance – CIM_JobQueue Reference

1031 This command form is for the `show` verb applied to a single instance. This command form corresponds to
1032 the `show` command issued against CIM_HostedJobDestination where the reference specified is to an
1033 instance of CIM_JobQueue. An instance of CIM_JobQueue is referenced by exactly one instance of
1034 CIM_HostedJobDestination. Therefore, a single instance will be returned.

1035 6.10.2.1.1 Command Form

```
1036 show <CIM_HostedJobDestination single instance>
```

1037 6.10.2.1.2 CIM Requirements

1038 See CIM_HostedJobDestination in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
1039 mandatory properties.

1040 6.10.2.1.3 Behavior Requirements

1041 6.10.2.1.3.1 Preconditions

```
1042 1) $instance contains the instance of CIM_JobQueue  
1043    which is referenced by CIM_HostedJobDestination  
1044    &smShowAssociationInstances ( "CIM_HostedJobDestination",  
1045    $instance.getObjectPath() );  
1046    smEnd;
```

1047 6.10.2.2 Show a Single Instance – CIM_ComputerSystem Reference

1048 This command form is for the `show` verb applied to a single instance. This command form corresponds to
1049 the `show` command issued against CIM_HostedJobDestination where the reference specified is to an
1050 instance of CIM_ComputerSystem. An instance of CIM_ComputerSystem is referenced by exactly one
1051 instance of CIM_HostedJobDestination. Therefore, a single instance will be returned.

1052 6.10.2.2.1 Command Form

```
1053 show <CIM_HostedJobDestination single instance>
```

1054 6.10.2.2.2 CIM Requirements

1055 See CIM_HostedJobDestination in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
1056 mandatory properties.

1057 6.10.2.2.3 Behavior Requirements

1058 6.10.2.2.3.1 Preconditions

```
1059 1) $instance contains the instance of CIM_ComputerSystem which is referenced by  
1060 CIM_HostedJobDestination
```

1061 6.10.2.2.3.2 Pseudo Code

```
1062 &smShowAssociationInstances ( "CIM_HostedJobDestination", $instance.getObjectPath() );  
1063 smEnd;
```

1064 6.10.2.3 Show a Single Instance – Both References

1065 This command form is for the `show` verb applied to a single instance. This command form corresponds to
1066 the `show` command issued against CIM_HostedJobDestination where both references are specified and
1067 therefore the desired instance is unambiguously identified.

1068 6.10.2.3.1 Command Form

```
1069 show <CIM_HostedJobDestination single instance>
```

1070 6.10.2.3.2 CIM Requirements

1071 See CIM_HostedJobDestination in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
1072 mandatory properties.

1073 6.10.2.3.3 Behavior Requirements

1074 6.10.2.3.3.1 Preconditions

```
1075 1) $instanceA contains the instance of CIM_ComputerSystem which is referenced by  
1076 CIM_HostedJobDestination  
1077 2) $instanceB contains the instance of CIM_JobQueue which is referenced by  
1078 CIM_HostedJobDestination.
```

1079 6.10.2.3.3.2 Pseudo Code

```
1080 &smShowAssociationInstance ( "CIM_HostedJobDestination",  
1081 $instanceA.getObjectPath(),$instanceB.getObjectPath() );  
1082 smEnd;
```

1083 6.11 CIM_HostedService

1084 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

1085 Table 10 lists each SM CLP verb, the required level of support for the verb in conjunction with instances
1086 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
1087 verb and target. Table 10 is for informational purposes only; in case of a conflict between Table 10 and
1088 requirements detailed in the following sections, the text detailed in the following sections supersedes the
1089 information in Table 10.

1090

Table 10 – Command Verb Requirements for CIM_HostedService

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 6.11.2.
start	Not supported	
stop	Not supported	

1091 No mapping is defined for the following verbs for the specified target: create, delete, dump, load,
1092 reset, set, start, and stop.

1093 6.11.1 Ordering of Results

1094 When results are returned for multiple instances of CIM_HostedService, implementations shall utilize the
1095 following algorithm to produce the natural (that is, default) ordering:

- 1096 • Results for CIM_HostedService are unordered; therefore, no algorithm is defined.

1097 6.11.2 Show

1098 This section describes how to implement the `show` verb when applied to an instance of
1099 CIM_HostedService. Implementations shall support the use of the `show` verb with CIM_HostedService.

1100 The `show` command is used to display information about the CIM_HostedService instance or instances.

1101 6.11.2.1 Show Multiple Instances

1102 This command form is for the `show` verb applied to multiple instances. This command form corresponds
1103 to a `show` command issued against CIM_HostedService where only one reference is specified and the
1104 reference is to an instance of CIM_ComputerSystem.

1105 6.11.2.1.1 Command Form

1106 `show <CIM_HostedService multiple instances>`

1107 6.11.2.1.2 CIM Requirements

1108 See CIM_HostedService in the “CIM Elements” section of the [CLP Service Profile](#) for the list of mandatory
1109 properties.

1110 6.11.2.1.3 Behavior Requirements

1111 6.11.2.1.3.1 Preconditions

1112 1) `instance` contains the instance of CIM_ComputerSystem which is referenced by
1113 CIM_HostedService

1114 6.11.2.1.3.2 Pseudo Code

```
1115 &smShowAssociationInstances ( "CIM_HostedService", $instance.getObjectPath() );
1116 smEnd;
```

1117 6.11.2.2 Show a Single Instance – CIM_ProtocolService Reference

1118 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 1119 the `show` command issued against `CIM_HostedService` where the reference specified is to an instance of
 1120 `CIM_ProtocolService`. An instance of `CIM_ProtocolService` is referenced by exactly one instance of
 1121 `CIM_HostedService`. Therefore, a single instance will be returned.

1122 6.11.2.2.1 Command Form

```
1123 show <CIM_HostedService single instance>
```

1124 6.11.2.2.2 CIM Requirements

1125 See `CIM_HostedService` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of mandatory
 1126 properties.

1127 6.11.2.2.3 Behavior Requirements

1128 6.11.2.2.3.1 Preconditions

```
1129 1) $instance contains the instance of CIM_ProtocolService which is referenced by
1130 CIM_HostedService
```

1131 6.11.2.2.3.2 Pseudo Code

```
1132 &smShowAssociationInstances ( "CIM_HostedService", $instance.getObjectPath() );
1133 smEnd;
```

1134 6.11.2.3 Show a Single Instance – Both References

1135 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 1136 the `show` command issued against `CIM_HostedService` where both references are specified and
 1137 therefore the desired instance is unambiguously identified.

1138 6.11.2.3.1 Command Form

```
1139 show <CIM_HostedService single instance>
```

1140 6.11.2.3.2 CIM Requirements

1141 See `CIM_HostedService` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of mandatory
 1142 properties.

1143 6.11.2.3.3 Behavior Requirements

1144 6.11.2.3.3.1 Preconditions

```
1145 1) $instanceA contains the instance of CIM_ComputerSystem which is referenced by
1146 CIM_HostedService
1147 2) $instanceB contains the instance of CIM_ProtocolService which is referenced by
1148 CIM_HostedService.
```

1149 **6.11.2.3.3.2 Pseudo Code**

```

1150 &smShowAssociationInstance ( "CIM_HostedService",
1151     $instanceA.getObjectPath(), $instanceB.getObjectPath() );
1152 smEnd;

```

1153 **6.12 CIM_JobDestinationJobs**

1154 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

1155 Table 11 lists each SM CLP verb, the required level of support for the verb in conjunction with instances
 1156 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 1157 verb and target. Table 11 is for informational purposes only; in case of a conflict between Table 11 and
 1158 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 1159 information in Table 11.

1160 **Table 11 – Command Verb Requirements for CIM_JobDestinationJobs**

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 6.12.2.
start	Not supported	
stop	Not supported	

1161 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 1162 `reset`, `set`, `start`, and `stop`.

1163 **6.12.1 Ordering of Results**

1164 When results are returned for multiple instances of `CIM_JobDestinationJobs`, implementations shall utilize
 1165 the following algorithm to produce the natural (that is, default) ordering:

- 1166 • Results for `CIM_JobDestinationJobs` are unordered; therefore, no algorithm is defined.

1167 **6.12.2 Show**

1168 This section describes how to implement the `show` verb when applied to an instance of
 1169 `CIM_JobDestinationJobs`. Implementations shall support the use of the `show` verb with
 1170 `CIM_JobDestinationJobs`.

1171 The `show` command is used to display information about the `CIM_JobDestinationJobs` instance or
 1172 instances.

1173 6.12.2.1 Show a Single Instance – CIM_ConcreteJob Reference

1174 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 1175 the `show` command issued against `CIM_JobDestinationJobs` where the reference specified is to an
 1176 instance of `CIM_ConcreteJob`. A single instance of `CIM_JobQueue` is associated with each instance of a
 1177 `CIM_ConcreteJob`. Therefore, a single instance will be returned.

1178 6.12.2.1.1 Command Form

```
1179 show <CIM_JobDestinationJobs single instance>
```

1180 6.12.2.1.2 CIM Requirements

1181 See `CIM_JobDestinationJobs` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
 1182 mandatory properties.

1183 6.12.2.1.3 Behavior Requirements

1184 6.12.2.1.3.1 Preconditions

```
1185 $instance contains the instance of CIM_ConcreteJob which is referenced by  

  1186 CIM_JobDestinationJobs
```

1187 6.12.2.1.3.2 Pseudo Code

```
1188 &smShowAssociationInstances ( "CIM_OwningJobElement", $instance.getObjectPath() );  

  1189 smEnd;
```

1190 6.12.2.2 Show Multiple Instances – CIM_JobQueue Reference

1191 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 1192 the `show` command issued against `CIM_JobDestinationJobs` where the reference specified is to an
 1193 instance of `CIM_JobQueue`. A single instance of `CIM_JobQueue` is associated with multiple instances of
 1194 `CIM_ConcreteJob`. Therefore, multiple instances may be returned.

1195 6.12.2.2.1 Command Form

```
1196 show <CIM_JobDestinationJobs multiple instances>
```

1197 6.12.2.2.2 CIM Requirements

1198 See `CIM_JobDestinationJobs` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
 1199 mandatory properties.

1200 6.12.2.2.3 Behavior Requirements

1201 6.12.2.2.3.1 Preconditions

```
1202 $instance contains the instance of CIM_JobQueue which is referenced by  

  1203 CIM_JobDestinationJobs
```

1204 6.12.2.2.3.2 Pseudo Code

```
1205 &smShowAssociationInstances ( "CIM_JobDestinationJobs", $instance.getObjectPath() );  

  1206 smEnd;
```

1207 6.12.2.3 Show a Single Instance – Both References

1208 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 1209 the `show` command issued against `CIM_JobDestinationJobs` where both references are specified and
 1210 therefore the desired instance is unambiguously identified.

1211 6.12.2.3.1 Command Form

```
1212 show <CIM_JobDestinationJobs single instance>
```

1213 6.12.2.3.2 CIM Requirements

1214 See `CIM_JobDestinationJobs` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of
 1215 mandatory properties.

1216 6.12.2.3.3 Behavior Requirements

1217 6.12.2.3.3.1 Preconditions

- ```
1218 1) $instanceA contains the instance of CIM_ConcreteJob which is referenced by
1219 CIM_JobDestinationJobs
1220 2) $instanceB contains the instance of CIM_JobQueue which is referenced by
1221 CIM_JobDestinationJobs.
```

##### 1222 6.12.2.3.3.2 Pseudo Code

```
1223 &smShowAssociationInstance ("CIM_JobDestinationJobs",
1224 $instanceA.getObjectPath(), $instanceB.getObjectPath());
1225 smEnd;
```

## 1226 6.13 CIM\_JobQueue

1227 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

1228 Table 12 lists each SM CLP verb, the required level of support for the verb in conjunction with instances  
 1229 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the  
 1230 verb and target. Table 12 is for informational purposes only; in case of a conflict between Table 12 and  
 1231 requirements detailed in the following sections, the text detailed in the following sections supersedes the  
 1232 information in Table 12.

1233 **Table 12 – Command Verb Requirements for CIM\_JobQueue**

| 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 6.13.2. |
| start        | Not supported |             |
| stop         | Not supported |             |

1234 No mapping is defined for the following verbs for the specified target: `dump` and `load`.



### 1235 6.13.1 Ordering of Results

1236 When results are returned for multiple instances of CIM\_JobQueue, implementations shall utilize the  
1237 following algorithm to produce the natural (that is, default) ordering:

- 1238 • Results for CIM\_JobQueue are unordered; therefore, no algorithm is defined.

### 1239 6.13.2 Show

1240 This section describes how to implement the `show` verb when applied to an instance of CIM\_JobQueue.  
1241 Implementations shall support the use of the `show` verb with CIM\_JobQueue.

1242 The `show` verb is used to display information about the CLP Service's Job Queue.

#### 1243 6.13.2.1 Show a Single Instance

1244 This command form is for the `show` verb applied to a single instance of CIM\_JobQueue.

##### 1245 6.13.2.1.1 Command Form

```
1246 show <CIM_JobQueue single instance>
```

##### 1247 6.13.2.1.2 CIM Requirements

1248 See CIM\_JobQueue in the "CIM Elements" section of the [CLP Service Profile](#) for the list of mandatory  
1249 properties.

##### 1250 6.13.2.1.3 Behavior Requirements

###### 1251 6.13.2.1.3.1 Preconditions

```
1252 #all is true if the all option was specified with the command; otherwise, #all is
1253 false
1254 $instance=<CIM_JobQueue Single Instance>
1255 #propertylist[] = null;
1256 if (false == #all) {
1257 #propertylist[] = { //all mandatory non-key properties }
1258 }
1259 &smShowInstance($instance.getObjectPath(), #propertylist[]);
```

#### 1260 6.13.2.2 Show Multiple Instances

1261 This command form is for the `show` verb applied to a single instance of CIM\_JobQueue.

##### 1262 6.13.2.2.1 Command Form

```
1263 show <CIM_JobQueue single instance>
```

##### 1264 6.13.2.2.2 CIM Requirements

1265 See CIM\_JobQueue in the "CIM Elements" section of the [CLP Service Profile](#) for the list of mandatory  
1266 properties.

##### 1267 6.13.2.2.3 Behavior Requirements

###### 1268 6.13.2.2.3.1 Preconditions

```
1269 1) $containerInstance contains the instance of CIM_ComputerSystem for which we are
1270 displaying scoped CIM_JobQueue instances.
```

```
1271 2) #all is true if the all option was specified with the command; otherwise, #all is
1272 false
```

### 1273 6.13.2.2.3.2 Pseudo Code

```
1274 #propertylist[] = null;
1275 if (false == #all) {
1276 #propertylist[] = { //all mandatory non-key properties }
1277 }
1278 &smShowInstances ("CIM_JobQueue", "CIM_HostedJobDestination",
1279 $containerInstance.getObjectPath(), #propertylist[]);
1280 &smEnd;
```

## 1281 6.14 CIM\_OwningJobElement

1282 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

1283 Table 13 lists each SM CLP verb, the required level of support for the verb in conjunction with instances  
 1284 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the  
 1285 verb and target. Table 13 is for informational purposes only; in case of a conflict between Table 13 and  
 1286 requirements detailed in the following sections, the text detailed in the following sections supersedes the  
 1287 information in Table 13.

1288 **Table 13 – Command Verb Requirements for CIM\_OwningJobElement**

| 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 6.14.2. |
| start        | Not supported |             |
| stop         | Not supported |             |

1289 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,  
 1290 `reset`, `set`, `start`, and `stop`.

### 1291 6.14.1 Ordering of Results

1292 When results are returned for multiple instances of `CIM_OwningJobElement`, implementations shall utilize  
 1293 the following algorithm to produce the natural (that is, default) ordering:

- 1294 • Results for `CIM_OwningJobElement` are unordered; therefore, no algorithm is defined.

### 1295 6.14.2 Show

1296 This section describes how to implement the `show` verb when applied to an instance of  
 1297 `CIM_OwningJobElement`. Implementations shall support the use of the `show` verb with  
 1298 `CIM_OwningJobElement`.

1299 The `show` command is used to display information about the `CIM_OwningJobElement` instance or  
1300 instances.

### 1301 **6.14.2.1 Show a Single Instance – CIM\_ConcreteJob Reference**

1302 This command form is for the `show` verb applied to a single instance. This command form corresponds to  
1303 the `show` command issued against `CIM_OwningJobElement` where the reference specified is to an  
1304 instance of `CIM_ConcreteJob`. A single instance of `CIM_ProtocolService` is associated with each instance  
1305 of a `CIM_ConcreteJob`. Therefore, a single instance will be returned.

#### 1306 **6.14.2.1.1 Command Form**

```
1307 show <CIM_OwningJobElement single instance>
```

#### 1308 **6.14.2.1.2 CIM Requirements**

1309 See `CIM_OwningJobElement` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
1310 mandatory properties.

#### 1311 **6.14.2.1.3 Behavior Requirements**

##### 1312 **6.14.2.1.3.1 Preconditions**

```
1313 1) $instance contains the instance of CIM_ConcreteJob which is referenced by
1314 CIM_OwningJobElement
```

##### 1315 **6.14.2.1.3.2 Pseudo Code**

```
1316 &smShowAssociationInstances ("CIM_OwningJobElement", $instance.getObjectPath());
1317 smEnd;
```

### 1318 **6.14.2.2 Show Multiple Instances – CIM\_ProtocolService Reference**

1319 This command form is for the `show` verb applied to a single instance. This command form corresponds to  
1320 the `show` command issued against `CIM_OwningJobElement` where the reference specified is to an  
1321 instance of `CIM_ProtocolService`. A single instance of `CIM_ProtocolService` is associated with multiple  
1322 instances of `CIM_ConcreteJob`. Therefore, multiple instances may be returned.

#### 1323 **6.14.2.2.1 Command Form**

```
1324 show <CIM_OwningJobElement multiple instances>
```

#### 1325 **6.14.2.2.2 CIM Requirements**

1326 See `CIM_OwningJobElement` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
1327 mandatory properties.

#### 1328 **6.14.2.2.3 Behavior Requirements**

##### 1329 **6.14.2.2.3.1 Preconditions**

```
1330 1) $instance contains the instance of CIM_ProtocolService which is referenced by
1331 CIM_OwningJobElement
```

##### 1332 **6.14.2.2.3.2 Pseudo Code**

```
1333 &smShowAssociationInstances ("CIM_OwningJobElement", $instance.getObjectPath());
1334 smEnd;
```

### 1335 6.14.2.3 Show a Single Instance – Both References

1336 This command form is for the `show` verb applied to a single instance. This command form corresponds to  
 1337 the `show` command issued against `CIM_OwningJobElement` where both references are specified and  
 1338 therefore the desired instance is unambiguously identified.

#### 1339 6.14.2.3.1 Command Form

```
1340 show <CIM_OwningJobElement single instance>
```

#### 1341 6.14.2.3.2 CIM Requirements

1342 See `CIM_OwningJobElement` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
 1343 mandatory properties.

#### 1344 6.14.2.3.3 Behavior Requirements

##### 1345 6.14.2.3.3.1 Preconditions

- 1346 1) `$instanceA` contains the instance of `CIM_ConcreteJob` which is referenced by
- 1347 `CIM_OwningJobElement`
- 1348 2) `$instanceB` contains the instance of `CIM_ProtocolService` which is referenced by
- 1349 `CIM_OwningJobElement`.

##### 1350 6.14.2.3.3.2 Pseudo Code

```
1351 &smShowAssociationInstance ("CIM_OwningJobElement",

 1352 $instanceA.getObjectPath(), $instanceB.getObjectPath());

 1353 smEnd;
```

## 1354 6.15 CIM\_ProtocolService

1355 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

1356 Table 14 lists each SM CLP verb, the required level of support for the verb in conjunction with instances  
 1357 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the  
 1358 verb and target. Table 14 is for informational purposes only; in case of a conflict between Table 14 and  
 1359 requirements detailed in the following sections, the text detailed in the following sections supersedes the  
 1360 information in Table 14.

1361 **Table 14 – Command Verb Requirements for `CIM_ProtocolService`**

| Command Verb | Requirement   | Comments    |
|--------------|---------------|-------------|
| create       | Not supported |             |
| delete       | Not supported |             |
| dump         | Not supported |             |
| load         | Not supported |             |
| reset        | May           | See 6.15.2. |
| set          | May           | See 6.15.3. |
| show         | Shall         | See 6.15.4. |
| start        | May           | See 6.15.5. |
| stop         | May           | See 6.15.6. |

1362 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, and `load`.

### 1363 6.15.1 Ordering of Results

1364 When results are returned for multiple instances of CIM\_ProtocolService, implementations shall utilize the  
1365 following algorithm to produce the natural (that is, default) ordering:

- 1366 • Results for CIM\_ProtocolService are unordered; therefore, no algorithm is defined.

### 1367 6.15.2 Reset

1368 This section describes how to implement the `reset` verb when applied to an instance of  
1369 CIM\_ProtocolService. Implementations may support the use of the `reset` verb with  
1370 CIM\_ProtocolService.

1371 The `reset` verb is used to initiate a reset of the CIM\_ProtocolService.

#### 1372 6.15.2.1 Reset a Single Instance

1373 This command form is for the initiation of a reset action against a single instance of the  
1374 CIM\_ProtocolService. The mapping is implemented as an invocation of the `RequestStateChange()`  
1375 method on the instance.

##### 1376 6.15.2.1.1 Command Form

```
1377 reset <CIM_ProtocolService single instance>
```

##### 1378 6.15.2.1.2 CIM Requirements

```
1379 uint16 EnabledState;
1380 uint16 RequestedState;
1381 uint32 EnabledLogicalElement.RequestStateChange (
1382 [IN] uint16 RequestedState,
1383 [OUT] REF CIM_ConcreteJob Job,
1384 [IN] datetime TimeoutPeriod);
```

##### 1385 6.15.2.1.3 Behavior Requirements

```
1386 $instance=<CIM_ProtocolService Single Instance>
1387 smResetRSC($instance.GetObjectPath());
1388 &smEnd;
```

### 1389 6.15.3 Set

1390 This section describes how to implement the `set` verb when it is applied to an instance of  
1391 CIM\_ProtocolService. Implementations may support the use of the `set` verb with CIM\_ProtocolService.

1392 The `set` verb is used to modify descriptive properties of the CIM\_ProtocolService instance.

#### 1393 6.15.3.1 General Usage of Set for a Single Property

1394 This command form corresponds to the general usage of the `set` verb to modify a single property of a  
1395 target instance. This is the most common case.

1396 The requirement for supporting modification of a property using this command form shall be equivalent to  
1397 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined  
1398 in the [CLP Service Profile](#).

##### 1399 6.15.3.1.1 Command Form

```
1400 set <CIM_ProtocolService Single Instance> <propertyname>=<propertyvalue>
```

### 1401 6.15.3.1.2 CIM Requirements

1402 See CIM\_ProtocolService in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
1403 modifiable properties.

### 1404 6.15.3.1.3 Behavior Requirements

```
1405 $instance = <CIM_ProtocolService Single Instance>
1406 #propertyName[] = {<propertyname>};
1407 #propertyValues[] = {<propertyvalue>};
1408 &smSetInstance($instance, #propertyName[], #propertyValues[]);
1409 &smEnd;
```

### 1410 6.15.3.2 General Usage of Set for Multiple Properties

1411 This command form corresponds to the general usage of the set verb to modify multiple properties of a  
1412 target instance where there is not an explicit relationship between the properties. This is the most  
1413 common case.

1414 The requirement for supporting modification of a property using this command form shall be equivalent to  
1415 the requirement for supporting modification of the property using the ModifyInstance operation as defined  
1416 in the [CLP Service Profile](#).

#### 1417 6.15.3.2.1 Command Form

```
1418 set <CIM_ProtocolService Single Instance> <propertyname1>=<propertyvalue1>
1419 <propertynamen>=<propertyvaluen>
```

### 1420 6.15.3.2.2 CIM Requirements

1421 See CIM\_ProtocolService in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
1422 mandatory properties.

### 1423 6.15.3.2.3 Behavior Requirements

```
1424 $instance = <CIM_ProtocolService Single Instance>
1425 #propertyName[] = {<propertyname>};
1426
1427 for #i < n
1428 {
1429 #propertyName[#i] = <propertyname#i>
1430 #propertyValues[#i] = <propertyvalue#i>
1431 }
1432
1433 &smSetInstance($instance, #propertyName[], #propertyValues[]);
1434 &smEnd;
```

## 1435 6.15.4 Show

1436 This section describes how to implement the show verb when applied to an instance of  
1437 CIM\_ProtocolService. Implementations shall support the use of the show verb with CIM\_ProtocolService.

1438 The show verb is used to display information about the CIM\_ProtocolService.

### 1439 6.15.4.1 Show a Single Instance

1440 This command form is for the show verb applied to a single instance of CIM\_ProtocolService.

1441 **6.15.4.1.1 Command Form**1442 `show <CIM_ProtocolService single instance>`1443 **6.15.4.1.2 CIM Requirements**1444 See CIM\_ProtocolService in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
1445 mandatory properties.1446 **6.15.4.1.3 Behavior Requirements**1447 **6.15.4.1.3.1 Preconditions**1448 #all is true if the all option was specified with the command; otherwise, #all is  
1449 false  
1450 \$instance=<CIM\_ProtocolService *Single Instance*>1451 **6.15.4.1.3.2 Pseudo Code**1452 #propertylist[] = null;  
1453 if (false == #all) {  
1454 #propertylist[] = { //all mandatory non-key properties };  
1455 }  
1456  
1457 &smShowInstance(\$instance.getObjectPath(), #propertylist[]);  
1458 &smEnd;1459 **6.15.4.2 Show Multiple Instances**1460 This command form is for the show verb applied to multiple instances of CIM\_ProtocolService. This  
1461 command form corresponds to UFsT-based selection within a scoping system.1462 **6.15.4.2.1 Command Form**1463 `show <CIM_ProtocolService multiple instances>`1464 **6.15.4.2.2 CIM Requirements**1465 See CIM\_ProtocolService in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
1466 mandatory properties.1467 **6.15.4.2.3 Behavior Requirements**1468 **6.15.4.2.3.1 Preconditions**1469 1) \$containerInstance contains the instance of CIM\_ComputerSystem for which we are  
1470 displaying scoped instances of the CIM\_ProtocolService. The CLP Service Profile  
1471 requires that the CIM\_ProtocolService instance be associated with its scoping  
1472 system via an instance of the CIM\_HostedService association.  
1473 2) #all is true if the all option was specified with the command; otherwise, #all is  
1474 false1475 **6.15.4.2.3.2 Pseudo Code**1476 #propertylist[] = null;  
1477 if (false == #all) {  
1478 #propertylist[] = { //all mandatory non-key properties };  
1479 }  
1480  
1481 &smShowInstances ( "CIM\_ProtocolService", "CIM\_HostedService",  
1482 \$containerInstance.getObjectPath(), #propertylist[] );  
1483 &smEnd;

1484 **6.15.5 Start**

1485 This section describes how to implement the `start` verb when applied to an instance of  
 1486 `CIM_ProtocolService`. Implementations may support the use of the `start` verb with  
 1487 `CIM_ProtocolService`.

1488 The `start` verb is used to enable the `CIM_ProtocolService`.

1489 **6.15.5.1 Start a Single Instance**

1490 This command form is for the `start` verb applied to a single instance of `CIM_ProtocolService`.

1491 **6.15.5.1.1 Command Form**

```
1492 start <CIM_ProtocolService single instance>
```

1493 **6.15.5.1.2 CIM Requirements**

```
1494 uint16 EnabledState;
1495 uint16 RequestedState;
1496 uint32 EnabledLogicalElement.RequestStateChange (
1497 [IN] uint16 RequestedState,
1498 [OUT] REF CIM_ConcreteJob Job,
1499 [IN] datetime TimeoutPeriod);
```

1500 **6.15.5.1.3 Behavior Requirements**

```
1501 $instance=<CIM_ProtocolService Single Instance>
1502 smStartRSC($instance.getObjectPath());
1503 &smEnd;
```

1504 **6.15.6 Stop**

1505 This section describes how to implement the `stop` verb when applied to an instance of  
 1506 `CIM_ProtocolService`. Implementations may support the use of the `stop` verb with `CIM_ProtocolService`.

1507 The `stop` verb is used to disable the `CIM_ProtocolService`.

1508 **6.15.6.1 Stop a Single Instance**

1509 This command form is for the `stop` verb applied to a single instance of `CIM_ProtocolService`.

1510 **6.15.6.1.1 Command Form**

```
1511 stop <CIM_ProtocolService single instance>
```

1512 **6.15.6.1.2 CIM Requirements**

```
1513 uint16 EnabledState;
1514 uint16 RequestedState;
1515 uint32 EnabledLogicalElement.RequestStateChange (
1516 [IN] uint16 RequestedState,
1517 [OUT] REF CIM_ConcreteJob Job,
1518 [IN] datetime TimeoutPeriod);
```

1519 **6.15.6.1.3 Behavior Requirements**

```
1520 $instance=<CIM_ProtocolService Single Instance>
1521 smStopRSC($instance.getObjectPath());
1522 &smEnd;
```



## 1523 6.16 CIM\_ProvidesEndpoint

1524 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

1525 Table 15 lists each SM CLP verb, the required level of support for the verb in conjunction with instances  
 1526 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the  
 1527 verb and target. Table 15 is for informational purposes only; in case of a conflict between Table 15 and  
 1528 requirements detailed in the following sections, the text detailed in the following sections supersedes the  
 1529 information in Table 15.

1530 **Table 15 – Command Verb Requirements for CIM\_ProvidesEndpoint**

| 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 6.16.2. |
| start        | Not supported |             |
| stop         | Not supported |             |

1531 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,  
 1532 `reset`, `set`, `start`, and `stop`.

### 1533 6.16.1 Ordering of Results

1534 When results are returned for multiple instances of `CIM_ProvidesEndpoint`, implementations shall utilize  
 1535 the following algorithm to produce the natural (that is, default) ordering:

- 1536 Results for `CIM_ProvidesEndpoint` are unordered; therefore, no algorithm is defined.

### 1537 6.16.2 Show

1538 This section describes how to implement the `show` verb when applied to an instance of  
 1539 `CIM_ProvidesEndpoint`. Implementations shall support the use of the `show` verb with  
 1540 `CIM_ProvidesEndpoint`.

1541 The `show` command is used to display information about the `CIM_ProvidesEndpoint` instance or  
 1542 instances.

#### 1543 6.16.2.1 Show a Single Instance – CIM\_CLPProtocolEndpoint Reference

1544 This command form is for the `show` verb applied to a single instance. This command form corresponds to  
 1545 the `show` command issued against `CIM_ProvidesEndpoint` where the reference specified is to an  
 1546 instance of `CIM_CLPProtocolEndpoint`. A single instance of `CIM_ProtocolService` is associated with each  
 1547 instance of a `CIM_CLPProtocolEndpoint`. Therefore, a single instance will be returned.

##### 1548 6.16.2.1.1 Command Form

1549 `show <CIM_ProvidesEndpoint single instance>`

**1550 6.16.2.1.2 CIM Requirements**

1551 See CIM\_ProvidesEndpoint in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
1552 mandatory properties.

**1553 6.16.2.1.3 Behavior Requirements****1554 6.16.2.1.3.1 Preconditions**

1555 1) \$instance contains the instance of CIM\_CLPProtocolEndpoint which is referenced by  
1556 CIM\_ProvidesEndpoint

**1557 6.16.2.1.3.2 Pseudo Code**

```
1558 &smShowAssociationInstances ("CIM_ProvidesEndpoint", $instance.getObjectPath());
1559 smEnd;
```

**1560 6.16.2.2 Show Multiple Instances – CIM\_ProtocolService Reference**

1561 This command form is for the `show` verb applied to a single instance. This command form corresponds to  
1562 the `show` command issued against CIM\_ProvidesEndpoint where the reference specified is to an  
1563 instance of CIM\_ProtocolService. A single instance of CIM\_ProtocolService is associated with multiple  
1564 instances of a CIM\_CLPProtocolEndpoint. Therefore, multiple instances may be returned.

**1565 6.16.2.2.1 Command Form**

```
1566 show <CIM_ProvidesEndpoint multiple instances>
```

**1567 6.16.2.2.2 CIM Requirements**

1568 See CIM\_ProvidesEndpoint in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
1569 mandatory properties.

**1570 6.16.2.2.3 Behavior Requirements****1571 6.16.2.2.3.1 Preconditions**

1572 1) \$instance contains the instance of CIM\_ProtocolService which is referenced by  
1573 CIM\_ProvidesEndpoint

**1574 6.16.2.2.3.2 Pseudo Code**

```
1575 &smShowAssociationInstances ("CIM_ProvidesEndpoint", $instance.getObjectPath());
1576 smEnd;
```

**1577 6.16.2.3 Show a Single Instance – Both References**

1578 This command form is for the `show` verb applied to a single instance. This command form corresponds to  
1579 the `show` command issued against CIM\_ProvidesEndpoint where both references are specified and  
1580 therefore the desired instance is unambiguously identified.

**1581 6.16.2.3.1 Command Form**

```
1582 show <CIM_ProvidesEndpoint single instance>
```

**1583 6.16.2.3.2 CIM Requirements**

1584 See CIM\_ProvidesEndpoint in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
1585 mandatory properties.

1586 **6.16.2.3.3 Behavior Requirements**

1587 **6.16.2.3.3.1 Preconditions**

- 1588 1) \$instanceA contains the instance of CIM\_CLPProtocolEndpoint which is referenced by  
 1589 CIM\_ProvidesEndpoint  
 1590 2) \$instanceB contains the instance of CIM\_ProtocolService which is referenced by  
 1591 CIM\_ProvidesEndpoint.

1592 **6.16.2.3.3.2 Pseudo Code**

```
1593 &smShowAssociationInstance ("CIM_ProvidesEndpoint",
1594 $instanceA.getObjectPath(),$instanceB.getObjectPath());
1595 smEnd;
```

1596 **6.17 CIM\_ServiceAffectsElement**

1597 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

1598 Table 16 lists each SM CLP verb, the required level of support for the verb in conjunction with instances  
 1599 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the  
 1600 verb and target. Table 16 is for informational purposes only; in case of a conflict between Table 16 and  
 1601 requirements detailed in the following sections, the text detailed in the following sections supersedes the  
 1602 information in Table 16.

1603 **Table 16 – Command Verb Requirements for CIM\_ServiceAffectsElement**

| 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 6.17.2. |
| start        | Not supported |             |
| stop         | Not supported |             |

1604 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,  
 1605 `reset`, `set`, `start`, and `stop`.

1606 **6.17.1 Ordering of Results**

1607 When results are returned for multiple instances of CIM\_ServiceAffectsElement, implementations shall  
 1608 utilize the following algorithm to produce the natural (that is, default) ordering:

- 1609 • Results for CIM\_ServiceAffectsElement are unordered; therefore, no algorithm is defined.

1610 **6.17.2 Show**

1611 This section describes how to implement the `show` verb when applied to an instance of  
 1612 CIM\_ServiceAffectsElement. Implementations shall support the use of the `show` verb with  
 1613 CIM\_ServiceAffectsElement.

1614 The `show` command is used to display information about the `CIM_ServiceAffectsElement` instance or  
1615 instances.

### 1616 **6.17.2.1 Show Multiple Instances – CIM\_ProtocolService Reference**

1617 This command form is for the `show` verb applied to a single instance. This command form corresponds to  
1618 the `show` command issued against `CIM_ServiceAffectsElement` where the reference specified is to an  
1619 instance of `CIM_ProtocolService`. Multiple instances of `CIM_ServiceAffectsElement` reference an instance  
1620 of `CIM_ProtocolService`.

#### 1621 **6.17.2.1.1 Command Form**

```
1622 show < CIM_ServiceAffectsElement multiple instances >
```

#### 1623 **6.17.2.1.2 CIM Requirements**

1624 See `CIM_ServiceAffectsElement` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
1625 mandatory properties.

#### 1626 **6.17.2.1.3 Behavior Requirements**

##### 1627 **6.17.2.1.3.1 Preconditions**

```
1628 1) $instance contains the instance of CIM_ProtocolService which is referenced by
1629 CIM_ServiceAffectsElement
```

##### 1630 **6.17.2.1.3.2 Pseudo Code**

```
1631 &smShowAssociationInstances ("CIM_ServiceAffectsElement",
1632 $instance.getObjectPath());
1633 smEnd;
```

### 1634 **6.17.2.2 Show a Single Instance – CIM\_JobQueue Reference**

1635 This command form is for the `show` verb applied to a single instance. This command form corresponds to  
1636 the `show` command issued against `CIM_ServiceAffectsElement` where the reference specified is to an  
1637 instance of `CIM_JobQueue`. A single instance of `CIM_ProtocolService` is associated with each instance of  
1638 a `CIM_JobQueue`. Therefore, a single instance will be returned.

#### 1639 **6.17.2.2.1 Command Form**

```
1640 show < CIM_ServiceAffectsElement single instance >
```

#### 1641 **6.17.2.2.1.1 CIM Requirements**

1642 See `CIM_ServiceAffectsElement` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
1643 mandatory properties.

#### 1644 **6.17.2.2.2 Behavior Requirements**

##### 1645 **6.17.2.2.2.1 Preconditions**

```
1646 1) $instance contains the instance of CIM_JobQueue which is referenced by
1647 CIM_ServiceAffectsElement
```

##### 1648 **6.17.2.2.2.2 Pseudo Code**

```
1649 &smShowAssociationInstances ("CIM_ServiceAffectsElement",
1650 $instance.getObjectPath());
1651 smEnd;
```

### 1652 **6.17.2.3 Show a Single Instance – CIM\_AdminDomain Reference**

1653 This command form is for the `show` verb applied to a single instance. This command form corresponds to  
 1654 the `show` command issued against `CIM_ServiceAffectsElement` where the reference specified is to an  
 1655 instance of `CIM_AdminDomain`. A single instance of `CIM_ProtocolService` is associated with each  
 1656 instance of a `CIM_AdminDomain`. Therefore, a single instance will be returned.

#### 1657 **6.17.2.3.1 Command Form**

```
1658 show < CIM_ServiceAffectsElement single instance >
```

#### 1659 **6.17.2.3.2 CIM Requirements**

1660 See `CIM_AffectsElement` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
 1661 mandatory properties.

#### 1662 **6.17.2.3.3 Behavior Requirements**

##### 1663 **6.17.2.3.3.1 Preconditions**

```
1664 $instance contains the instance of CIM_AdminDomain which is referenced by

 1665 CIM_ServiceAffectsElement
```

##### 1666 **6.17.2.3.3.2 Pseudo Code**

```
1667 &smShowAssociationInstances ("CIM_ServiceAffectsElement",

 1668 $instance.getObjectPath());

 1669 smEnd;
```

### 1670 **6.17.2.4 Show a Single Instance – Both References**

1671 This command form is for the `show` verb applied to a single instance. This command form corresponds to  
 1672 the `show` command issued against `CIM_ServiceAffectsElement` where both references are specified and  
 1673 therefore the desired instance is unambiguously identified.

#### 1674 **6.17.2.4.1 Command Form**

```
1675 show < CIM_ServiceAffectsElement single instance >
```

#### 1676 **6.17.2.4.2 CIM Requirements**

1677 See `CIM_AffectsElement` in the “CIM Elements” section of the [CLP Service Profile](#) for the list of  
 1678 mandatory properties.

#### 1679 **6.17.2.4.3 Behavior Requirements**

##### 1680 **6.17.2.4.3.1 Preconditions**

```
1681 1) $instanceA contains the instance of CIM_JobQueue or CIM_AdminDomain that is

 1682 referenced by CIM_ServiceAffectsElement

 1683 2) $instanceB contains the instance of CIM_ProtocolService which is referenced by

 1684 CIM_ServiceAffectsElement.
```

##### 1685 **6.17.2.4.3.2 Pseudo Code**

```
1686 &smShowAssociationInstance ("CIM_ServiceAffectsElement",

 1687 $instanceA.getObjectPath(), $instanceB.getObjectPath());

 1688 smEnd;
```

1689

**ANNEX A**  
(informative)**Change Log**

| Version | Date    | Author | Description           |
|---------|---------|--------|-----------------------|
| 1.0.0   | 7/29/09 |        | DMTF Standard Release |
|         |         |        |                       |

1690  
1691  
1692  
1693  
1694

1695