



1
2
3
4

Document Number: DSP0818

Date: 2009-07-29

Version: 1.0.0

5 **DHCP Client Profile SM CLP Command Mapping**
6 **Specification**

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

10

11 Copyright Notice

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

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

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27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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

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

33

CONTENTS

34	Foreword	5
35	Introduction	6
36	1 Scope	7
37	2 Normative References.....	7
38	2.1 Approved References	7
39	2.2 Other References.....	7
40	3 Terms and Definitions.....	7
41	4 Symbols and Abbreviated Terms.....	8
42	5 Recipes.....	9
43	6 Mappings.....	9
44	6.1 CIM_DHCPCapabilities.....	9
45	6.2 CIM_DHCPSettingData	11
46	6.3 CIM_DHCPProtocolEndpoint.....	13
47	6.4 CIM_ElementCapabilities	17
48	6.5 CIM_ElementSettingData	19
49	6.6 CIM_HostedAccessPoint	22
50	6.7 CIM_RemoteAccessAvailableToElement.....	25
51	6.8 CIM_RemoteServiceAccessPoint.....	27
52	6.9 CIM_SAPSAPDependency.....	31
53	ANNEX A (informative) Change Log.....	34
54		

55 Tables

56	Table 1 – Command Verb Requirements for CIM_DHCPCapabilities.....	10
57	Table 2 – Command Verb Requirements for CIM_DHCPSettingData	12
58	Table 3 – Command Verb Requirements for CIM_DHCPProtocolEndpoint.....	14
59	Table 4 – Command Verb Requirements for CIM_ElementCapabilities	17
60	Table 5 – Command Verb Requirements for CIM_ElementSettingData	19
61	Table 6 – Command Verb Requirements for CIM_HostedAccessPoint	22
62	Table 7 – Command Verb Requirements for CIM_RemoteAccessAvailableToElement.....	25
63	Table 8 – Command Verb Requirements for CIM_RemoteServiceAccessPoint.....	28
64	Table 9 – Command Verb Requirements for CIM_SAPSAPDependency.....	31
65		

67

Foreword

68 The *DHCP Client Profile SM CLP Command Mapping Specification* (DSP0818) was prepared by the
69 Server Management Working Group.

70 **Conventions**

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

73 **Acknowledgements**

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

76 **Editor:**

- 77 • Aaron Merkin – IBM

78 **Contributors:**

- 79 • Jon Hass – Dell
- 80 • Khachatur Papanyan – Dell
- 81 • Enoch Suen – Dell
- 82 • Jeff Hilland – HP
- 83 • Christina Shaw – HP
- 84 • Aaron Merkin – IBM
- 85 • Perry Vincent – Intel
- 86 • John Leung – Intel

87

88

Introduction

89 This document defines the SM CLP mapping for CIM elements described in the [DHCP Client Profile](#). The
90 information in this specification, combined with *SM CLP-to-CIM Common Mapping Specification V1.0*
91 ([DSP0216](#)), is intended to be sufficient to implement SM CLP commands relevant to the classes,
92 properties and methods described in the [DHCP Client Profile](#) using CIM operations.

93 The target audience for this specification is implementers of the SM CLP support for the [DHCP Client](#)
94 [Profile](#).

95

96
97

DHCP Client Profile SM CLP Command Mapping Specification

1 Scope

99 This specification contains the requirements for an implementation of the SM CLP to provide access to,
100 and implement the behaviors of, the [DHCP Client Profile](#).

2 Normative References

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

2.1 Approved References

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

108 DMTF DSP1037, *DHCP Client Profile 1.0*,
109 http://www.dmtf.org/standards/published_documents/DSP1037_1.0.pdf

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

2.2 Other References

113 ISO/IEC Directives, Part 2, [Rules for the structure and drafting of International Standards](#)

3 Terms and Definitions

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

3.1

can

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

3.2

cannot

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

3.3

conditional

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

- 126 **3.4**
127 **mandatory**
128 indicates requirements to be followed strictly in order to conform to the document and from which no
129 deviation is permitted
- 130 **3.5**
131 **may**
132 indicates a course of action permissible within the limits of the document
- 133 **3.6**
134 **need not**
135 indicates a course of action permissible within the limits of the document
- 136 **3.7**
137 **optional**
138 indicates a course of action permissible within the limits of the document
- 139 **3.8**
140 **shall**
141 indicates requirements to be followed strictly in order to conform to the document and from which no
142 deviation is permitted
- 143 **3.9**
144 **shall not**
145 indicates requirements to be followed strictly in order to conform to the document and from which no
146 deviation is permitted
- 147 **3.10**
148 **should**
149 indicates that among several possibilities, one is recommended as particularly suitable, without
150 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 151 **3.11**
152 **should not**
153 indicates that a certain possibility or course of action is deprecated but not prohibited
- 154 **3.12**
155 **unspecified**
156 indicates that this profile does not define any constraints for the referenced CIM element or operation

157 **4 Symbols and Abbreviated Terms**

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

- 159 **4.1**
160 **CIM**
161 Common Information Model
- 162 **4.2**
163 **CLP**
164 Common Information Model

165 **4.3**
166 **DMTF**
167 Distributed Management Task Force

168 **4.4**
169 **IETF**
170 Internet Engineering Task Force

171 **4.5**
172 **RFC**
173 Request for Comment

174 **4.6**
175 **SM**
176 Server Management

177 **4.7**
178 **SMI-S**
179 Storage Management Initiative Specification

180 **4.8**
181 **SNIA**
182 Storage Networking Industry Association

183 **5 Recipes**

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

- 186 • smStartRSC()
- 187 • smStopRSC()
- 188 • smResetRSC()
- 189 • smShowInstance()
- 190 • smShowInstances()
- 191 • smSetInstance()
- 192 • smShowAssociationInstances()
- 193 • smShowAssociationInstance()

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

195 **6 Mappings**

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

199 **6.1 CIM_DHCPCapabilities**

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

201 Table 1 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 202 the target class, and when appropriate, a cross-reference to the section detailing the mapping for the verb
 203 and target. Table 1 is for informational purposes only; in case of a conflict between this table and
 204 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 205 information in Table 1.

206 **Table 1 – Command Verb Requirements for CIM_DHCPCapabilities**

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	

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

209 6.1.1 Ordering of Results

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

- 212 • Results for CIM_DHCPCapabilities are unordered; therefore, no algorithm is defined.

213 6.1.2 Show

214 The `show` verb is used to display information about instances of CIM_DHCPCapabilities. Implementations
 215 shall support the use of the `show` verb with CIM_DHCPCapabilities.

216 6.1.2.1 Show Command Form for Single Instance

217 This command form is used when the `show` verb applies to a single instance of CIM_DHCPCapabilities.

218 6.1.2.1.1 Command Form

```
219 show <CIM_DHCPCapabilities single instance>
```

220 6.1.2.1.2 CIM Requirements

221 See CIM_DHCPCapabilities in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
 222 mandatory properties.

223 6.1.2.1.3 Behavior Requirements

224 6.1.2.1.3.1 Preconditions

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

226 6.1.2.1.3.2 Pseudo Code

```

227 $instance=<CIM_DHCPCapabilities single instance>
228     #propertylist[] = null;
229     if (false == #all)
230     {
231         #propertylist[] = {/all mandatory non-key properties};
232     }
233 &smShowInstance($instance.getObjectPath(), #propertylist[]);
234 &smEnd;

```

235 6.1.2.2 Show Command Form for Multiple Instances

236 This command form is used when the `show` verb applies to multiple instances of `CIM_DHCPCapabilities`.
 237 This command form corresponds to UFT-based selection within a capabilities collection.

238 6.1.2.2.1 Command Form

```

239 show <CIM_DHCPCapabilities multiple instances>

```

240 6.1.2.2.2 CIM Requirements

241 See `CIM_DHCPCapabilities` in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
 242 mandatory properties.

243 6.1.2.2.3 Behavior Requirements

244 6.1.2.2.3.1 Preconditions

245 `$containerInstance` contains the instance of `CIM_ConcreteCollection` for which contained
 246 `CIM_Capabilities` instances are displayed. `CIM_Capabilities` instances are addressed via the aggregating
 247 instance of `CIM_ConcreteCollection`

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

249 6.1.2.2.3.2 Pseudo Code

```

250 #propertylist[] = null;
251     if (false == #all)
252     {
253         #propertylist[] = {/all non-key properties};
254     }
255 &smShowInstances ( "CIM_DHCPCapabilities", "CIM_MemberOfCollection",
256     $containerInstance.getObjectPath(), #propertylist[] );
257 &smEnd;

```

258 6.2 CIM_DHCPSettingData

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

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

265

Table 2 – Command Verb Requirements for CIM_DHCPSettingData

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	

266 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 267 `reset`, `set`, `start`, and `stop`.

268 6.2.1 Ordering of Results

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

- 271 • Results for CIM_DHCPSettingData are unordered; therefore, no algorithm is defined.

272 6.2.2 Show

273 The `show` verb is used to display information about instances of CIM_DHCPSettingData.
 274 Implementations shall support the use of the `show` verb with CIM_USBRedirectionService.

275 6.2.2.1 Show Command Form for Single Instance

276 This command form is used when the `show` verb applies to a single instance of CIM_DHCPSettingData.

277 6.2.2.1.1 Command Form

```
278 show <CIM_DHCPSettingData single instance>
```

279 6.2.2.1.2 CIM Requirements

280 See CIM_DHCPSettingData in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
 281 mandatory properties.

282 6.2.2.1.3 Behavior Requirements

283 6.2.2.1.3.1 Preconditions

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

285 6.2.2.1.3.2 Pseudo Code

```

286 $instance=<CIM_DHCPSettingData single instance>
287     #propertylist[] = null;
288     if (false == #all)
289     {
290         #propertylist[] = { //all mandatory non-key properties }
291     }
292 &smShowInstance($instance.getObjectPath(), #propertylist[]);
293 &smEnd;

```

294 6.2.2.2 Show Command Form for Multiple Instances

295 This command form is used when the `show` verb applies to multiple instances of `CIM_DHCPSettingData`.
 296 This command form corresponds to UFsT-based selection within a scoping system.

297 6.2.2.2.1 Command Form

```

298 show <CIM_DHCPSettingData multiple instances>

```

299 6.2.2.2.2 CIM Requirements

300 See `CIM_DHCPSettingData` in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
 301 mandatory properties.

302 6.2.2.2.3 Behavior Requirements

303 6.2.2.2.3.1 Preconditions

304 `$containerInstance` represents the instance of `CIM_IPAssignmentSettingData` for which related
 305 `CIM_DHCPSettingData` instances are displayed.

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

307 6.2.2.2.3.2 Pseudo Code

```

308 #propertylist[] = null;
309     if (false == #all)
310     {
311         #propertylist[] = { //all mandatory non-key properties }
312     }
313 &smShowInstances ( "CIM_DHCPSettingData", "CIM_OrderedComponent",
314     $containerInstance.getObjectPath(), #propertylist[] );
315 &smEnd;

```

316 6.3 CIM_DHCPProtocolEndpoint

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

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

323

Table 3 – Command Verb Requirements for CIM_DHCPProtocolEndpoint

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	Not supported	

324 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 325 `reset`, `start`, and `stop`.

326 6.3.1 Ordering of Results

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

- 329 • Results for `CIM_DHCPProtocolEndpoint` are unordered; therefore, no algorithm is defined.

330 6.3.2 Set

331 The `set` verb is used to modify descriptive properties of the `CIM_DHCPProtocolEndpoint` instance.
 332 Implementations may support the use of the `set` verb with `CIM_DHCPProtocolEndpoint`.

333 6.3.2.1 General Usage of Set for a Single Property

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

336 The requirement for supporting modification of a property using this command form shall be equivalent to
 337 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined
 338 in the [DHCP Client Profile](#).

339 6.3.2.1.1 Command Form

```
340 set <CIM_DHCPProtocolEndpoint single object> <propertyname>=<propertyvalue>
```

341 6.3.2.1.2 CIM Requirements

342 See `CIM_DHCPProtocolEndpoint` in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
 343 any modifiable properties.

344 6.3.2.1.3 Behavior Requirements

```
345 $instance=<CIM_DHCPProtocolEndpoint single object>
346     #propertyName[] = {<propertyname>};
347     #propertyValues[] = {<propertyvalue>};
348 &smSetInstance($instance, #propertyName[], #propertyValues[]);
349 &smEnd;
```

350 6.3.2.2 General Usage of Set for Multiple Properties

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

354 The requirement for supporting modification of a property using this command form shall be equivalent to
 355 the requirement for supporting modification of the property using the ModifyInstance operation as defined
 356 in the [DHCP Client Profile](#).

357 6.3.2.2.1 Command Form

```
358 set <CIM_DHCPProtocolEndpoint multiple objects> <propertyname1>=<propertyvalue1>  
359 <propertynamen>=<propertyvaluen>
```

360 6.3.2.2.2 CIM Requirements

361 See CIM_DHCPProtocolEndpoint in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
 362 mandatory properties.

363 6.3.2.2.3 Behavior Requirements

```
364 $instance=<CIM_DHCPProtocolEndpoint multiple objects>  
365 #propertyNames[] = {<propertyname>};  
366     for #i < n  
367     {  
368         #propertyNames[#i] = <propertname#i>  
369         #propertyValues[#i] = <propertyvalue#i>  
370     }  
371 &smSetInstance($instance, #propertyNames[], #propertyValues[]);  
372 &smEnd;
```

373 6.3.3 Show

374 The `show` verb is used to display information about instances of CIM_DHCPProtocolEndpoint.
 375 Implementations shall support the use of the `show` verb with CIM_DHCPProtocolEndpoint.

376 6.3.3.1 Show Command Form for Single Instance

377 This command form is used when the `show` verb applies to a single instance of
 378 CIM_DHCPProtocolEndpoint.

379 6.3.3.1.1 Command Form

```
380 show <CIM_DHCPProtocolEndpoint single instance>
```

381 6.3.3.1.2 CIM Requirements

382 See CIM_DHCPProtocolEndpoint in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
 383 mandatory properties.

384 6.3.3.1.3 Behavior Requirements

385 6.3.3.1.3.1 Preconditions

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

387 **6.3.3.1.3.2 Pseudo Code**

```

388 $instance=<CIM_DHCPProtocolEndpoint single instance>
389     #propertylist[] = null;
390     if (false == #all)
391     {
392         #propertylist[] = { //all mandatory non-key properties };
393     }
394 &smShowInstance($instance.getObjectPath(), #propertylist[]);
395 &smEnd;

```

396 **6.3.3.2 Show Command Form for Multiple Instances**

397 This command form is used when the `show` verb applies to multiple instances of
398 CIM_DHCPProtocolEndpoint. This command form corresponds to UFsT-based selection within a scoping
399 system.

400 **6.3.3.2.1 Command Form**

```

401 show <CIM_DHCPProtocolEndpoint multiple instances>

```

402 **6.3.3.2.2 CIM Requirements**

403 See CIM_DHCPProtocolEndpoint in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
404 mandatory properties.

405 **6.3.3.2.3 Behavior Requirements**406 **6.3.3.2.3.1 Preconditions**

407 \$containerInstance contains the instance of CIM_ComputerSystem for which scoped
408 CIM_DHCPProtocolEndpoint instances are displayed. The [DHCP Client Profile](#) requires that the
409 CIM_DHCPProtocolEndpoint instance be associated with its scoping system via an instance of the
410 CIM_HostedAccessPoint association.

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

412 **6.3.3.2.3.2 Pseudo Code**

```

413 #propertylist[] = null;
414     if (false == #all)
415     {
416         #propertylist[] = { //all mandatory non-key properties };
417     }
418 &smShowInstances ( "CIM_DHCPProtocolEndpoint", "CIM_HostedAccessPoint",
419     $containerInstance.getObjectPath(), #propertylist[] );
420 &smEnd;

```


421 **6.4 CIM_ElementCapabilities**

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

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

428 **Table 4 – 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 6.4.2.
Start	Not supported	
Stop	Not supported	

429 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 430 `reset`, `set`, `start`, and `stop`.

431 **6.4.1 Ordering of Results**

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

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

435 **6.4.2 Show**

436 The `show` verb is used to display information about instances of `CIM_ElementCapabilities`.
 437 Implementations shall support the use of the `show` verb with `CIM_ElementCapabilities`.

438 **6.4.2.1 Show Command Form for Multiple Instances – CIM_DHCPCapabilities Reference**

439 This command form is used when the `show` verb applies to multiple instances. This command form
 440 corresponds to a `show` command issued against instances of `CIM_ElementCapabilities` where only one
 441 reference is specified and the reference is to an instance of `CIM_DHCPCapabilities`.

442 **6.4.2.1.1 Command Form**

443 `show <CIM_ElementCapabilities multiple instances>`

444 **6.4.2.1.2 CIM Requirements**

445 See `CIM_ElementCapabilities` in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
 446 mandatory properties.

447 6.4.2.1.3 Behavior Requirements

448 6.4.2.1.3.1 Preconditions

449 \$instance contains the instance of CIM_DHCPCapabilities which is referenced by
450 CIM_ElementCapabilities.

451 6.4.2.1.3.2 Pseudo Code

```
452 &smShowAssociationInstances ( "CIM_ElementCapabilities", $instance.getObjectPath() );  
453 &smEnd;
```

454 6.4.2.2 Show Command Form for Single Instance – CIM_DHCPProtocolEndpoint Reference

455 This command form is used when the `show` verb applies to a single instance. This command form
456 corresponds to a `show` command issued against instances of CIM_ElementCapabilities where the
457 reference specified is to an instance of CIM_DHCPProtocolEndpoint. An instance of
458 CIM_DHCPProtocolEndpoint is referenced by exactly one instance of CIM_ElementCapabilities;
459 therefore, a single instance will be returned.

460 6.4.2.2.1 Command Form

```
461 show <CIM_ElementCapabilities single instance>
```

462 6.4.2.2.2 CIM Requirements

463 See CIM_ElementCapabilities in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
464 mandatory properties.

465 6.4.2.2.3 Behavior Requirements

466 6.4.2.2.3.1 Preconditions

467 \$instance represents the instance of CIM_DHCPProtocolEndpoint, which is referenced by
468 CIM_ElementCapabilities.

469 6.4.2.2.3.2 Pseudo Code

```
470 &smShowAssociationInstances ( "CIM_ElementCapabilities", $instance.getObjectPath() );  
471 &smEnd;
```

472 6.4.2.3 Show Command Form for Single Instance – Both References

473 This command form is used when the `show` verb applies to a single instance. This command form
474 corresponds to a `show` command issued against instances of CIM_ElementCapabilities where both
475 references are specified; therefore, the desired instance is unambiguously identified.

476 6.4.2.3.1 Command Form

```
477 show <CIM_ElementCapabilities single instance>
```

478 6.4.2.3.2 CIM Requirements

479 See CIM_ElementCapabilities in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
480 mandatory properties.

481 6.4.2.3.3 Behavior Requirements

482 6.4.2.3.3.1 Preconditions

483 \$instanceA contains the instance of CIM_DHCPCapabilities which is referenced by
484 CIM_ElementCapabilities.

485 \$instanceB contains the instance of CIM_DHCPProtocolEndpoint which is referenced by
486 CIM_ElementCapabilities.

487 6.4.2.3.3.2 Pseudo Code

```
488 &smShowAssociationInstance ( "CIM_ElementCapabilities", $instanceA.getObjectPath(),
489     $instanceB.getObjectPath() );
490 &smEnd;
```

491 6.5 CIM_ElementSettingData

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

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

498 **Table 5 – 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	Not supported	
Show	Shall	See 6.5.2.
Start	Not supported	
Stop	Not supported	

499 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
500 `reset`, `set`, `start`, and `stop`.

501 6.5.1 Ordering of Results

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

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

505 6.5.2 Show

506 The `show` verb is used to display information about instances of `CIM_ElementSettingData`.
507 Implementations shall support the use of the `show` verb with `CIM_ElementSettingData`.

508 6.5.2.1 Show Command Form for Multiple Instances – `CIM_DHCPProtocolEndpoint` Reference

509 This command form is used when the `show` verb applies to multiple instances. This command form
510 corresponds to a `show` command issued against instances of `CIM_ElementSettingData` where only one
511 reference is specified and the reference is to an instance of `CIM_DHCPProtocolEndpoint`.

512 6.5.2.1.1 Command Form

```
513 show <CIM_ElementSettingData multiple instances>
```

514 6.5.2.1.2 CIM Requirements

515 See `CIM_ElementSettingData` in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
516 mandatory properties.

517 6.5.2.1.3 Behavior Requirements

518 6.5.2.1.3.1 Preconditions

519 `$instance` contains the instance of `CIM_DHCPProtocolEndpoint` which is referenced by
520 `CIM_ElementSettingData`.

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

522 6.5.2.1.3.2 Pseudo Code

```
523 #propertylist[] = null;  
524 if (#all == false)  
525     {  
526         #propertylist[] = { "IsCurrent" };  
527     }  
528 &smShowAssociationInstances ( "CIM_ElementSettingData", $instance.getObjectPath(),  
529     #propertylist[] );  
530 &smEnd;
```

531 6.5.2.2 Show Command Form for Multiple Instances – `CIM_DHCPSettingData` Reference

532 This command form is used when the `show` verb applies to multiple instances. This command form
533 corresponds to a `show` command issued against instances of `CIM_ElementSettingData` where only one
534 reference is specified and the reference is to an instance of `CIM_DHCPSettingData`.

535 6.5.2.2.1 Command Form

```
536 show <CIM_ElementSettingData multiple instances>
```

537 6.5.2.2.2 CIM Requirements

538 See `CIM_ElementSettingData` in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
539 mandatory properties.

540 6.5.2.2.3 Behavior Requirements

541 6.5.2.2.3.1 Preconditions

542 \$instance contains the instance of CIM_DHCPSettingData which is referenced by
543 CIM_ElementSettingData.

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

545 6.5.2.2.3.2 Pseudo Code

```
546 #propertylist[] = null;
547 if (#all == false)
548     {
549         #propertylist[] = { "IsCurrent" };
550     }
551 &smShowAssociationInstances ( "CIM_ElementSettingData", $instance.GetObjectPath(),
552     #propertylist[] );
553 &smEnd;
```

554 6.5.2.3 Show Command Form for Single Instance – Both References

555 This command form is used when the show verb applies to a single instance. This command form
556 corresponds to a show command issued against instances of CIM_ElementSettingData where both
557 references are specified; therefore, the desired instance is unambiguously identified.

558 6.5.2.3.1 Command Form

```
559 show <CIM_ElementSettingData single instance>
```

560 6.5.2.3.2 CIM Requirements

561 See CIM_ElementSettingData in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
562 mandatory properties.

563 6.5.2.3.3 Behavior Requirements

564 6.5.2.3.3.1 Preconditions

565 \$instanceA contains the instance of CIM_DHCPProtocolEndpoint which is referenced by
566 CIM_ElementSettingData.

567 \$instanceB contains the instance of CIM_DHCPSettingData which is referenced by
568 CIM_ElementSettingData.

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

570 6.5.2.3.3.2 Pseudo Code

```
571 #propertylist[] = null;
572 if (#all == false)
573     {
574         #propertylist[] = { "IsCurrent" };
575     }
576 &smShowAssociationInstance ( "CIM_ElementSettingData", $instanceA.GetObjectPath(),
577     $instanceB.GetObjectPath(), #propertylist[] );
578 &smEnd;
```

579 6.6 CIM_HostedAccessPoint

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

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

586 **Table 6 – 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 6.6.2.
Start	Not supported	
Stop	Not supported	

587 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 588 `reset`, `set`, `start`, and `stop`.

589 6.6.1 Ordering of Results

590 When results are returned for multiple instances of `CIM_HostedAccessPoint`, implementations shall utilize
 591 the following algorithm to produce the natural (that is, default) ordering.

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

593 6.6.2 Show

594 The `show` verb is used to display information about instances of `CIM_HostedAccessPoint`.
 595 Implementations shall support the use of the `show` verb with `CIM_HostedAccessPoint`.

596 6.6.2.1 Show Command Form for Multiple Instances – CIM_ComputerSystem Reference

597 This command form is used when the `show` verb applies to multiple instances. This command form
 598 corresponds to a `show` command issued against instances of `CIM_HostedAccessPoint` where only one
 599 reference is specified and the reference is to an instance of `CIM_ComputerSystem`.

600 6.6.2.1.1 Command Form

601 `show <CIM_HostedAccessPoint multiple instances>`

602 6.6.2.1.2 CIM Requirements

603 See `CIM_HostedAccessPoint` in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
 604 mandatory properties.

605 **6.6.2.1.3 Behavior Requirements**

606 **6.6.2.1.3.1 Preconditions**

607 `$instance` contains the instance of `CIM_ComputerSystem` which is referenced by
608 `CIM_HostedAccessPoint`.

609 **6.6.2.1.3.2 Pseudo Code**

```
610 &smShowAssociationInstances ( "CIM_HostedAccessPoint", $instance.GetObjectPath() );  
611 &smEnd;
```

612 **6.6.2.2 Show Command Form for Single Instance – CIM_DHCPProtocolEndpoint Reference**

613 This command form is used when the `show` verb applies to a single instance. This command form
614 corresponds to a `show` command issued against instances of `CIM_HostedAccessPoint` where the
615 reference specified is to an instance of `CIM_DHCPProtocolEndpoint`. An instance of
616 `CIM_DHCPProtocolEndpoint` is referenced by exactly one instance of `CIM_HostedAccessPoint`;
617 therefore, a single instance will be returned.

618 **6.6.2.2.1 Command Form**

```
619 show <CIM_HostedAccessPoint single instance>
```

620 **6.6.2.2.2 CIM Requirements**

621 See `CIM_HostedAccessPoint` in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
622 mandatory properties.

623 **6.6.2.2.3 Behavior Requirements**

624 **6.6.2.2.3.1 Preconditions**

625 `$instance` contains the instance of `CIM_DHCPProtocolEndpoint` which is referenced by
626 `CIM_HostedAccessPoint`.

627 **6.6.2.2.3.2 Pseudo Code**

```
628 &smShowAssociationInstances ( "CIM_HostedAccessPoint", $instance.GetObjectPath() );  
629 &smEnd;
```

630 **6.6.2.3 Show Command Form for Single Instance – CIM_RemoteServiceAccessPoint Reference**

631 This command form is used when the `show` verb applies to a single instance. This command form
632 corresponds to a `show` command issued against `CIM_HostedAccessPoint` where the reference specified
633 is to an instance of `CIM_RemoteServiceAccessPoint`. An instance of `CIM_RemoteServiceAccessPoint` is
634 referenced by exactly one instance of `CIM_HostedAccessPoint`; therefore, a single instance will be
635 returned.

636 **6.6.2.3.1 Command Form**

```
637 show <CIM_HostedAccessPoint single instance>
```

638 **6.6.2.3.2 CIM Requirements**

639 See `CIM_HostedAccessPoint` in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
640 mandatory properties.

641 6.6.2.3.3 Behavior Requirements

642 6.6.2.3.3.1 Preconditions

643 \$instance contains the instance of CIM_RemoteServiceAccessPoint which is referenced by
644 CIM_HostedAccessPoint.

645 6.6.2.3.3.2 Pseudo Code

```
646 &smShowAssociationInstances ( "CIM_HostedAccessPoint", $instance.GetObjectPath() );  
647 &smEnd;
```

648 6.6.2.4 Show Command Form for Single Instance – Both References (DHCPProtocolEndpoint)

649 This command form is used when the `show` verb applies to a single instance. This command form
650 corresponds to a `show` command issued against instances of CIM_HostedAccessPoint where both
651 references are specified; therefore, the desired instance is unambiguously identified.

652 6.6.2.4.1 Command Form

```
653 show <CIM_HostedAccessPoint single instance>
```

654 6.6.2.4.2 CIM Requirements

655 See CIM_HostedAccessPoint in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
656 mandatory properties.

657 6.6.2.4.3 Behavior Requirements

658 6.6.2.4.3.1 Preconditions

659 \$instanceA contains the instance of CIM_ComputerSystem which is referenced by
660 CIM_HostedAccessPoint.

661 \$instanceB contains the instance of CIM_DHCPProtocolEndpoint which is referenced by
662 CIM_HostedAccessPoint.

663 6.6.2.4.3.2 Pseudo Code

```
664 &smShowAssociationInstance ( "CIM_HostedAccessPoint", $instanceA.GetObjectPath(),  
665     $instanceB.GetObjectPath() );  
666 &smEnd;
```

667 6.6.2.5 Show Command Form for Single Instance – Both References 668 (RemoteServiceAccessPoint)

669 This command form is used when the `show` verb applies to a single instance. This command form
670 corresponds to a `show` command issued against instances of CIM_HostedAccessPoint where both
671 references are specified; therefore, the desired instance is unambiguously identified.

672 6.6.2.5.1 Command Form

```
673 show <CIM_HostedAccessPoint single instance>
```

674 6.6.2.5.2 CIM Requirements

675 See CIM_HostedAccessPoint in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
676 mandatory properties.

677 **6.6.2.5.3 Behavior Requirements**

678 **6.6.2.5.3.1 Preconditions**

679 \$instanceA contains the instance of CIM_ComputerSystem which is referenced by
680 CIM_HostedAccessPoint.

681 \$instanceB contains the instance of CIM_RemoteServiceAccessPoint which is referenced by
682 CIM_HostedAccessPoint.

683 **6.6.2.5.3.2 Pseudo Code**

```
684 &smShowAssociationInstance ( "CIM_HostedAccessPoint", $instanceA.getObjectPath(),
685     $instanceB.getObjectPath() );
686 &smEnd;
```

687 **6.7 CIM_RemoteAccessAvailableToElement**

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

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

694 **Table 7 – Command Verb Requirements for CIM_RemoteAccessAvailableToElement**

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

695 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
696 `reset`, `set`, `start`, and `stop`.

697 **6.7.1 Ordering of Results**

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

- 700 • Results for CIM_RemoteAccessAvailableToElement are unordered; therefore, no algorithm is
701 defined.

702 6.7.2 Show

703 The `show` verb is used to display information about instances of `CIM_RemoteAccessAvailableToElement`.
704 Implementations shall support the use of the `show` verb with `CIM_RemoteAccessAvailableToElement`.

705 6.7.2.1 Show Command Form for Multiple Instances – `CIM_RemoteServiceAccessPoint` 706 Reference

707 This command form is used when the `show` verb applies to multiple instances. This command form
708 corresponds to a `show` command issued against instances of `CIM_RemoteAccessAvailableToElement`
709 where only one reference is specified and the reference is to an instance of
710 `CIM_RemoteServiceAccessPoint`.

711 6.7.2.1.1 Command Form

```
712 show <CIM_RemoteAccessAvailableToElement multiple instances>
```

713 6.7.2.1.2 CIM Requirements

714 See `CIM_RemoteAccessAvailableToElement` in the “CIM Elements” section of the [DHCP Client Profile](#) for
715 the list of mandatory properties.

716 6.7.2.1.3 Behavior Requirements

717 6.7.2.1.3.1 Preconditions

718 `$instance` contains the instance of `CIM_RemoteServiceAccessPoint` which is referenced by
719 `CIM_RemoteAccessAvailableToElement`.

720 There is only a single property and it is always returned.

721 6.7.2.1.3.2 Pseudo Code

```
722 &smShowAssociationInstances ( "CIM_RemoteAccessAvailableToElement",  
723     $instance.getObjectPath(), null );  
724 &smEnd;
```

725 6.7.2.2 Show Command Form for Multiple Instances – `CIM_DHCPProtocolEndpoint` Reference

726 This command form is used when the `show` verb applies to multiple instances. This command form
727 corresponds to a `show` command issued against instances of `CIM_RemoteAccessAvailableToElement`
728 where the reference specified is to an instance of `CIM_DHCPProtocolEndpoint`. The [DHCP Client Profile](#)
729 allows the implementation to model the DHCP servers discovered by the client in addition to the DHCP
730 Service that actually provides the configuration; therefore, it is possible for there to be multiple
731 `CIM_RemoteAccessAvailableToElement` associations that reference the `CIM_DHCPProtocolEndpoint`
732 instance.

733 6.7.2.2.1 Command Form

```
734 show <CIM_RemoteAccessAvailableToElement multiple instances>
```

735 6.7.2.2.2 CIM Requirements

736 See `CIM_RemoteAccessAvailableToElement` in the “CIM Elements” section of the [DHCP Client Profile](#) for
737 the list of mandatory properties.

738 6.7.2.2.3 Behavior Requirements

739 6.7.2.2.3.1 Preconditions

740 \$instance contains the instance of CIM_DHCPProtocolEndpoint which is referenced by
741 CIM_RemoteAccessAvailableToElement.

742 There is only a single property and it is always returned.

743 6.7.2.2.3.2 Pseudo Code

```
744 &smShowAssociationInstances ( "CIM_RemoteAccessAvailableToElement",
745     $instance.GetObjectPath(), null );
746 &smEnd;
```

747 6.7.2.3 Show Command Form for Single Instance – Both References

748 This command form is used when the `show` verb applies to a single instance. This command form
749 corresponds to a `show` command issued against instances of `CIM_RemoteAccessAvailableToElement`
750 where both references are specified; therefore, the desired instance is unambiguously identified.

751 6.7.2.3.1 Command Form

```
752 show <CIM_RemoteAccessAvailableToElement single instance>
```

753 6.7.2.3.2 CIM Requirements

754 See `CIM_RemoteAccessAvailableToElement` in the “CIM Elements” section of the [DHCP Client Profile](#) for
755 the list of mandatory properties.

756 6.7.2.3.3 Behavior Requirements

757 6.7.2.3.3.1 Preconditions

758 \$instanceA represents the referenced instance of `CIM_RemoteServiceAccessPoint` which is referenced
759 by `CIM_RemoteAccessAvailableToElement`.

760 \$instanceB represents the referenced instance of `CIM_DHCPProtocolEndpoint` which is referenced by
761 `CIM_RemoteAccessAvailableToElement`.

762 There is only a single property and it is always returned.

763 6.7.2.3.3.2 Pseudo Code

```
764 &smShowAssociationInstance ( "CIM_RemoteAccessAvailableToElement",
765     $instanceA.GetObjectPath(), $instanceB.GetObjectPath(), null );
766 &smEnd;
```

767 6.8 CIM_RemoteServiceAccessPoint

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

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

774

Table 8 – Command Verb Requirements for CIM_RemoteServiceAccessPoint

Command Verb	Requirement	Comments
Create	Not supported	
Delete	Not supported	
Dump	Not supported	
Load	Not supported	
Reset	Not supported	
Set	May	See 6.8.2.
Show	Shall	See 6.8.3.
Start	Not supported	
Stop	Not supported	

775 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
776 `reset`, `start`, and `stop`.

777 6.8.1 Ordering of Results

778 When results are returned for multiple instances of `CIM_RemoteServiceAccessPoint`, implementations
779 shall utilize the following algorithm to produce the natural (that is, default) ordering.

- 780 • Results for `CIM_RemoteServiceAccessPoint` are unordered; therefore, no algorithm is defined.

781 6.8.2 Set

782 The `set` verb is used to modify descriptive properties of the `CIM_RemoteServiceAccessPoint` instance.
783 Implementations may support the use of the `set` verb with `CIM_RemoteServiceAccessPoint`.

784 6.8.2.1 General Usage of Set for a Single Property

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

787 The requirement for supporting modification of a property using this command form shall be equivalent to
788 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined
789 in the [DHCP Client Profile](#).

790 6.8.2.1.1 Command Form

```
791 set <CIM_RemoteServiceAccessPoint single object> <propertyname>=<propertyvalue>
```

792 6.8.2.1.2 CIM Requirements

793 See `CIM_RemoteServiceAccessPoint` in the “CIM Elements” section of the [DHCP Client Profile](#) for the list
794 of modifiable properties.

795 6.8.2.1.3 Behavior Requirements

```
796 $instance=<CIM_RemoteServiceAccessPoint single object>
797     #propertyName[] = {<propertyname>};
798     #propertyValues[] = {<propertyvalue>};
799 &smSetInstance($instance, #propertyName[], #propertyValues[]);
800 &smEnd;
```

801 6.8.2.2 General Usage of Set for Multiple Properties

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

805 The requirement for supporting modification of a property using this command form shall be equivalent to
806 the requirement for supporting modification of the property using the ModifyInstance operation as defined
807 in the [DHCP Client Profile](#).

808 6.8.2.2.1 Command Form

```
809 set <CIM_RemoteServiceAccessPoint multiple objects> <propertyname1>=<propertyvalue1>  
810 <propertynamen>=<propertyvaluen>
```

811 6.8.2.2.2 CIM Requirements

812 See CIM_RemoteServiceAccessPoint in the “CIM Elements” section of the [DHCP Client Profile](#) for the list
813 of modifiable properties.

814 6.8.2.2.3 Behavior Requirements

```
815 $instance=<CIM_RemoteServiceAccessPoint multiple objects>  
816 #propertyNames[] = {<propertyname>};  
817 for #i < n  
818 {  
819     #propertyNames[#i] = <propertyname#i>  
820     #propertyValues[#i] = <propertyvalue#i>  
821 }  
822 &smSetInstance($instance, #propertyNames[], #propertyValues[]);  
823 &smEnd;
```

824 6.8.3 Show

825 The `show` verb is used to display information about instances of CIM_RemoteServiceAccessPoint.
826 Implementations shall support the use of the `show` verb with CIM_RemoteServiceAccessPoint.

827 6.8.3.1 Show Command Form for Single Instance

828 This command form is used when the `show` verb applies to a single instance of
829 CIM_RemoteServiceAccessPoint.

830 6.8.3.1.1 Command Form

```
831 show <CIM_RemoteServiceAccessPoint single instance>
```

832 6.8.3.1.2 CIM Requirements

833 See CIM_RemoteServiceAccessPoint in the “CIM Elements” section of the [DHCP Client Profile](#) for the list
834 of mandatory properties.

835 6.8.3.1.3 Behavior Requirements

836 6.8.3.1.3.1 Preconditions

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

838 **6.8.3.1.3.2 Pseudo Code**

```

839 $instance=<CIM_RemoteServiceAccessPoint single instance>
840     #propertylist[] = null;
841     if (false == #all)
842     {
843         #propertylist[] = {"AccessContext", "AccessInfo", "InfoFormat", "ElementName"};
844     }
845 &smShowInstance($instance.getObjectPath(), #propertylist[]);
846 &smEnd;

```

847 **6.8.3.2 Show Command Form for Multiple Instances**

848 This command form is used when the `show` verb applies to multiple instances of
849 `CIM_RemoteServiceAccessPoint`. This command form corresponds to UFT-based selection within a
850 scoping system.

851 **6.8.3.2.1 Command Form**

```

852 show <CIM_RemoteServiceAccessPoint multiple instances>

```

853 **6.8.3.2.2 CIM Requirements**

854 See `CIM_RemoteServiceAccessPoint` in the "CIM Elements" section of the [DHCP Client Profile](#) for the list
855 of mandatory properties.

856 **6.8.3.2.3 Behavior Requirements**857 **6.8.3.2.3.1 Preconditions**

858 `$containerInstance` contains the instance of `CIM_ComputerSystem` for which scoped
859 `CIM_RemoteServiceAccessPoint` instances are displayed. The [DHCP Client Profile](#) requires that the
860 `CIM_RemoteServiceAccessPoint` instance be associated with its scoping system via an instance of the
861 `CIM_HostedAccessPoint` association.

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

863 **6.8.3.2.3.2 Pseudo Code**

```

864 #propertylist[] = null;
865     if (false == #all)
866     {
867         #propertylist[] = {"AccessContext", "AccessInfo", "InfoFormat", "ElementName"};
868     }
869 &smShowInstances ( "CIM_RemoteServiceAccessPoint", "CIM_HostedAccessPoint",
870     $containerInstance.getObjectPath(), #propertylist[] );
871 &smEnd;

```

872 6.9 CIM_SAPSAPDependency

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

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

879 **Table 9 – Command Verb Requirements for CIM_SAPSAPDependency**

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	

880 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 881 `reset`, `set`, `start`, and `stop`.

882 6.9.1 Ordering of Results

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

- 885 • Results for CIM_SAPSAPDependency are unordered; therefore, no algorithm is defined.

886 6.9.2 Show

887 The `show` verb is used to display information about instances of CIM_SAPSAPDependency.
 888 Implementations shall support the use of the `show` verb with CIM_SAPSAPDependency.

889 6.9.2.1 Show Command Form for Single Instance – CIM_IPProtocolEndpoint Reference

890 This command form is used when the `show` verb applies to a single instance. This command form
 891 corresponds to a `show` command issued against instances of CIM_SAPSAPDependency where only one
 892 reference is specified and the reference is to an instance of CIM_IPProtocolEndpoint.

893 6.9.2.1.1 Command Form

894 `show <CIM_SAPSAPDependency single instance>`

895 6.9.2.1.2 CIM Requirements

896 See CIM_SAPSAPDependency in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
 897 mandatory properties.

898 6.9.2.1.3 Behavior Requirements

899 6.9.2.1.3.1 Preconditions

900 `$instance` represents the instance of CIM_IPProtocolEndpoint, which is referenced by
901 CIM_SAPSAPDependency.

902 6.9.2.1.3.2 Pseudo Code

```
903 &smShowAssociationInstances ("CIM_SAPSAPDependency", $instance.GetObjectPath() );  
904 &smEnd;
```

905 6.9.2.2 Show Command Form for Single Instance – CIM_DHCPProtocolEndpoint Reference

906 This command form is used when the `show` verb applies to a single instance. This command form
907 corresponds to a `show` command issued against instances of CIM_SAPSAPDependency where the
908 reference specified is to an instance of CIM_DHCPProtocolEndpoint. An instance of
909 CIM_DHCPProtocolEndpoint is referenced by exactly one instance of CIM_SAPSAPDependency;
910 therefore, a single instance will be returned.

911 6.9.2.2.1 Command Form

```
912 show <CIM_SAPSAPDependency single instance>
```

913 6.9.2.2.2 CIM Requirements

914 See CIM_SAPSAPDependency in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
915 mandatory properties.

916 6.9.2.2.3 Behavior Requirements

917 6.9.2.2.3.1 Preconditions

918 `$instance` represents the instance of CIM_DHCPProtocolEndpoint, which is referenced by
919 CIM_SAPSAPDependency.

920 6.9.2.2.3.2 Pseudo Code

```
921 &smShowAssociationInstances ("CIM_SAPSAPDependency", $instance.GetObjectPath() );  
922 &smEnd;
```

923 6.9.2.3 Show Command Form for Single Instance – Both References

924 This command form is used when the `show` verb applies to a single instance. This command form
925 corresponds to a `show` command issued against instances of CIM_SAPSAPDependency where both
926 references are specified; therefore, the desired instance is unambiguously identified.

927 6.9.2.3.1 Command Form

```
928 show <CIM_SAPSAPDependency single instance>
```

929 6.9.2.3.2 CIM Requirements

930 See CIM_SAPSAPDependency in the “CIM Elements” section of the [DHCP Client Profile](#) for the list of
931 mandatory properties.

932 6.9.2.3.3 Behavior Requirements**933 6.9.2.3.3.1 Preconditions**

934 \$instanceA represents the instance of CIM_IPProtocolEndpoint which is referenced by
935 CIM_SAPSAPDependency.

936 \$instanceB represents the instance of CIM_DHCPProtocolEndpoint which is referenced by
937 CIM_SAPSAPDependency.

938 6.9.2.3.3.2 Pseudo Code

```
939 &smShowAssociationInstance ( "CIM_SAPSAPDependency", $instanceA.getObjectPath(),  
940     $instanceB.getObjectPath() );  
941 &smEnd;
```

942

943
944
945
946
947

ANNEX A
(informative)

Change Log

Version	Date	Author	Description
1.0.0	07/29/09		DMTF Standard release.

948