



1
2
3
4

Document Number: DSP0837

Date: 2009-06-04

Version: 1.0.0

5 **USB Redirection Profile SM CLP Mapping**
6 **Specification**

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

10

11 Copyright notice

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

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

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27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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

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

33

CONTENTS

34 Foreword 5

35 Introduction 6

36 1 Scope 7

37 2 Normative References..... 7

38 2.1 Approved References 7

39 2.2 Other References..... 7

40 3 Terms and Definitions..... 7

41 4 Symbols and Abbreviated Terms..... 8

42 5 Recipes..... 9

43 6 Mappings..... 9

44 6.1 CIM_ElementCapabilities 9

45 6.2 CIM_HostedService 12

46 6.3 CIM_HostedAccessPoint 14

47 6.4 CIM_ServiceAffectsElement 17

48 6.5 CIM_SAPAvailableForElement..... 19

49 6.6 CIM_ServiceAccessBySAP 22

50 6.7 CIM_RemoteAccessAvailableToElement..... 24

51 6.8 CIM_BindsTo 27

52 6.9 CIM_LogicalIdentity 29

53 6.10 CIM_USBDevice 31

54 6.11 CIM_RemoteServiceAccessPoint..... 37

55 6.12 CIM_EnabledLogicalElementCapabilities..... 40

56 6.13 CIM_USBRedirectionCapabilities 42

57 6.14 CIM_USBRedirectionSAP 44

58 6.15 CIM_USBRedirectionService..... 57

59 ANNEX A (informative) Change Log 62

60

61 Tables

62 Table 1 – Command Verb Requirements for CIM_ElementCapabilities 10

63 Table 2 – Command Verb Requirements for CIM_HostedService 12

64 Table 3 – Command Verb Requirements for CIM_HostedAccessPoint 15

65 Table 4 – Command Verb Requirements for CIM_ServiceAffectsElement 17

66 Table 5 – Command Verb Requirements for CIM_SAPAvailableForElement..... 20

67 Table 6 – Command Verb Requirements for CIM_ServiceAccessBySAP 22

68 Table 7 – Command Verb Requirements for CIM_RemoteAccessAvailableToElement..... 25

69 Table 8 – Command Verb Requirements for CIM_BindsTo 27

70 Table 9 – Command Verb Requirements for CIM_LogicalIdentity 29

71 Table 10 – Command Verb Requirements for CIM_USBDevice 32

72 Table 11 – Command Verb Requirements for CIM_RemoteServiceAccessPoint..... 38

73 Table 12 – Command Verb Requirements for CIM_EnabledLogicalElementCapabilities..... 41

74 Table 13 – Command Verb Requirements for CIM_USBRedirectionCapabilities 43

75 Table 14 – Command Verb Requirements for CIM_USBRedirectionSAP 45

76 Table 15 – Command Verb Requirements for CIM_USBRedirectionService..... 58

77

79

Foreword

80 The *USB Redirection Profile SM CLP Mapping Specification* (DSP0837) was prepared by the Server
81 Management Working Group.

82 **Conventions**

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

85 **Acknowledgements**

86 The authors wish to acknowledge the following participants from the DMTF Server Management Working
87 Group:

- 88 • Joel Clark – Intel
- 89 • Aaron Merkin – IBM
- 90 • Jeff Hilland – HP
- 91 • Khachatur Papanyan – Dell

92

93

Introduction

94 This document defines the SM CLP mapping for CIM elements described in the [USB Redirection Profile](#).
95 The information in this specification, combined with the *SM CLP-to-CIM Common Mapping*
96 *Specification 1.0* ([DSP0216](#)), is intended to be sufficient to implement SM CLP commands relevant to the
97 classes, properties, and methods described in the [USB Redirection Profile](#) using CIM operations.

98 The target audience for this specification is implementers of the SM CLP support for the [USB Redirection](#)
99 [Profile](#).

100 USB Redirection Profile SM CLP Mapping Specification

101 1 Scope

102 This specification contains the requirements for an implementation of the SM CLP to provide access to,
103 and implement the behaviors of, the [USB Redirection Profile](#).

104 2 Normative References

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

108 2.1 Approved References

109 DMTF DSP1077, *USB Redirection Profile 1.0*,
110 http://www.dmtf.org/standards/published_documents/DSP1077_1.0.pdf

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

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

115 2.2 Other References

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

118 3 Terms and Definitions

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

120 3.1

121 **can**

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

123 3.2

124 **cannot**

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

126 3.3

127 **conditional**

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

130 3.4

131 **mandatory**

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

- 134 **3.5**
135 **may**
136 indicates a course of action permissible within the limits of the document
- 137 **3.6**
138 **need not**
139 indicates a course of action permissible within the limits of the document
- 140 **3.7**
141 **optional**
142 indicates a course of action permissible within the limits of the document
- 143 **3.8**
144 **shall**
145 indicates requirements to be followed strictly in order to conform to the document and from which no
146 deviation is permitted
- 147 **3.9**
148 **shall not**
149 indicates requirements to be followed strictly in order to conform to the document and from which no
150 deviation is permitted
- 151 **3.10**
152 **should**
153 indicates that among several possibilities, one is recommended as particularly suitable, without
154 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 155 **3.11**
156 **should not**
157 indicates that a certain possibility or course of action is deprecated but not prohibited

158 **4 Symbols and Abbreviated Terms**

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

- 160 **4.1**
161 **CIM**
162 Common Information Model
- 163 **4.2**
164 **CLP**
165 Command Line Protocol
- 166 **4.3**
167 **DMTF**
168 Distributed Management Task Force
- 169 **4.4**
170 **SM**
171 Server Management

- 172 **4.5**
 173 **SMI-S**
 174 Storage Management Initiative Specification
- 175 **4.6**
 176 **SNIA**
 177 Storage Networking Industry Association
- 178 **4.7**
 179 **UFsT**
 180 User Friendly selection Tag

181 **5 Recipes**

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

- 184 • smShowInstance()
- 185 • smShowInstances()
- 186 • smSetInstance()
- 187 • smShowAssociationInstance()
- 188 • smShowAssociationInstances()
- 189 • smRequestStateChange()
- 190 • smStartRSC()
- 191 • smStopRSC()

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

193 **6 Mappings**

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

197 **6.1 CIM_ElementCapabilities**

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

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

204

Table 1 – Command Verb Requirements for CIM_ElementCapabilities

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.1.2.
start	Not supported	
stop	Not supported	

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

207 6.1.1 Ordering of Results

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

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

211 6.1.2 Show

212 This section describes how to implement the `show` verb when applied to an instance of
213 CIM_ElementCapabilities. Implementations shall support the use of the `show` verb with
214 CIM_ElementCapabilities.

215 6.1.2.1 Show Command Form for a Single Instance – CIM_USBRedirectionService or 216 CIM_USBRedirectionSAP Reference

217 This command form is used when the `show` verb applies to a single instance. This command form
218 corresponds to the `show` command issued against instances of CIM_ElementCapabilities where only one
219 reference is specified and the reference is to the scoping instance of CIM_USBRedirectionService or
220 CIM_USBRedirectionSAP.

221 6.1.2.1.1 Command Form

```
222 show <CIM_ElementCapabilities single instance>
```

223 6.1.2.1.2 CIM Requirements

224 See CIM_ElementCapabilities in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
225 mandatory properties.

226 6.1.2.1.3 Behavior Requirements

227 6.1.2.1.3.1 Preconditions

228 `$instance` represents the instance of a CIM_USBRedirectionService or CIM_USBRedirectionSAP,
229 which is referenced by CIM_ElementCapabilities.

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

231 6.1.2.1.3.2 Pseudo Code

```
232 &smShowAssociationInstances ( "CIM_ElementCapabilities", $instance.getObjectPath() );
233 &smEnd;
```

234 6.1.2.2 Show Command Form for Multiple Instances – CIM_USBRedirectionCapabilities 235 Reference

236 This command form is used when the `show` verb applies to multiple instances. This command form
237 corresponds to the `show` command issued against instances of `CIM_ElementCapabilities` where only one
238 reference is specified and the reference is to the instance of `CIM_USBRedirectionCapabilities`.

239 6.1.2.2.1 Command Form

```
240 show <CIM_ElementCapabilities multiple instances>
```

241 6.1.2.2.2 CIM Requirements

242 See `CIM_ElementCapabilities` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
243 mandatory properties.

244 6.1.2.2.3 Behavior Requirements

245 6.1.2.2.3.1 Preconditions

246 `$instance` represents the instance of a `CIM_USBRedirectionServiceCapabilities`, which is referenced
247 by `CIM_ElementCapabilities`.

248 6.1.2.2.3.2 Pseudo Code

```
249 &smShowAssociationInstances ( "CIM_ElementCapabilities", $instance.getObjectPath(),
250     NULL );
251 &smEnd;
```

252 6.1.2.3 Show Command Form for Multiple Instances – CIM_EnabledLogicalElementCapabilities 253 Reference

254 This command form is used when the `show` verb applies to multiple instances. This command form
255 corresponds to the `show` command issued against instances of `CIM_ElementCapabilities` where only one
256 reference is specified and the reference is to the instance of `CIM_EnabledLogicalElementCapabilities`.

257 6.1.2.3.1 Command Form

```
258 show <CIM_ElementCapabilities multiple instances>
```

259 6.1.2.3.2 CIM Requirements

260 See `CIM_ElementCapabilities` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
261 mandatory properties.

262 6.1.2.3.3 Behavior Requirements

263 6.1.2.3.3.1 Preconditions

264 `$instance` represents the instance of a `CIM_EnabledLogicalElementCapabilities`, which is referenced
265 by `CIM_ElementCapabilities`.

266 **6.1.2.3.3.2 Pseudo Code**

```
267 &smShowAssociationInstances ( "CIM_ElementCapabilities", $instance.getObjectPath(),
268     NULL );
269 &smEnd;
```

270 **6.1.2.4 Show a Single Instance Target – Both References**

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

274 **6.1.2.4.1 Command Form**

```
275 show <CIM_ElementCapabilities single instance>
```

276 **6.1.2.4.2 CIM Requirements**

277 See `CIM_ElementCapabilities` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
 278 mandatory properties.

279 **6.1.2.4.3 Behavior Requirements**280 **6.1.2.4.3.1 Preconditions**

281 `$instanceA` represents the instance of a `CIM_USBRedirectionService` or `CIM_USBRedirectionSAP`
 282 which is referenced by the `CIM_ElementCapabilities`.

283 `$instanceB` represents the instance of a `CIM_USBRedirectionServiceCapabilities` or
 284 `CIM_EnabledLogicalElementCapabilities` which is referenced by the `CIM_ElementCapabilities`.

285 **6.1.2.4.3.2 Pseudo Code**

```
286 &smShowAssociationInstance ( "CIM_ElementCapabilities", $instanceA.getObjectPath(),
287     $instanceB.getObjectPath() );
288 &smEnd;
```

289 **6.2 CIM_HostedService**

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

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

296 **Table 2 – 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	

Command Verb	Requirement	Comments
set	Not supported	
show	Shall	See 6.2.2.
start	Not supported	
stop	Not supported	

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

299 6.2.1 Ordering of Results

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

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

303 6.2.2 Show

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

306 6.2.2.1 Show Command Form for Multiple Instances – `CIM_ComputerSystem` Reference

307 This command form is used when the `show` verb applies to multiple instances. This command form
308 corresponds to the `show` command issued against instances of `CIM_HostedService` where only one
309 reference is specified and the reference is to an instance of `CIM_ComputerSystem`.

310 6.2.2.1.1 Command Form

```
311 show <CIM_HostedService multiple instances>
```

312 6.2.2.1.2 CIM Requirements

313 See `CIM_HostedService` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
314 mandatory properties.

315 6.2.2.1.3 Behavior Requirements

316 6.2.2.1.3.1 Preconditions

317 `$instance` represents the instance of `CIM_ComputerSystem`, which is referenced by
318 `CIM_HostedService`.

319 6.2.2.1.3.2 Pseudo Code

```
320 &smShowAssociationInstances ( "CIM_HostedService", $instance.GetObjectPath() );  
321 &smEnd;
```

322 6.2.2.2 Show Command Form for a Single Instance – `CIM_USBRedirectionService` Reference

323 This command form is used when the `show` verb applies to a single instance. The command form
324 corresponds to the `show` verb issued against instances of `CIM_HostedService` where only one reference
325 is specified and the reference is to an instance of `CIM_USBRedirectionService`.

326 6.2.2.2.1 Command Form

```
327 show <CIM_HostedService single instance>
```

328 6.2.2.2.2 CIM Requirements

329 See CIM_HostedService in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
330 mandatory properties.

331 6.2.2.2.3 Behavior Requirements

332 6.2.2.2.3.1 Preconditions

333 \$instance represents the instance of CIM_USBRedirectionService, which is referenced by
334 CIM_HostedService.

335 6.2.2.2.3.2 Pseudo Code

```
336 &smShowAssociationInstances ( "CIM_HostedService", $instance.GetObjectPath() );  
337 &smEnd;
```

338 6.2.2.3 Show Command Form for a Single Instance – Both References

339 This command form is used when the show verb applies to a single instance. This command form
340 corresponds to the show command issued against CIM_HostedService where both references are
341 specified; therefore, the desired instance is unambiguously identified.

342 6.2.2.3.1 Command Form

```
343 show <CIM_HostedService single instance>
```

344 6.2.2.3.2 CIM Requirements

345 See CIM_HostedService in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
346 mandatory properties.

347 6.2.2.3.3 Behavior Requirements

348 6.2.2.3.3.1 Preconditions

349 \$instanceA represents the referenced instance of CIM_ComputerSystem through CIM_HostedService
350 association.

351 \$instanceB represents the other instance of CIM_USBRedirectionService which is referenced by
352 CIM_HostedService.

353 6.2.2.3.3.2 Pseudo Code

```
354 &smShowAssociationInstance ( "CIM_HostedService", $instanceA.GetObjectPath(),  
355     $instanceB.GetObjectPath() );  
356 &smEnd;
```

357 6.3 CIM_HostedAccessPoint

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

359 Table 3 lists each SM CLP verb, the required level of support for the verb in conjunction with the target
360 class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and
361 target. Table 3 is for informational purposes only; in case of a conflict between Table 3 and requirements

362 detailed in the following sections, the text detailed in the following sections supersedes the information in
 363 Table 3.

364 **Table 3 – 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 6.3.2.
start	Not supported	
stop	Not supported	

365 No mappings are defined for the following verbs for the specified target: *create*, *delete*, *dump*, *load*,
 366 *reset*, *set*, *start*, and *stop*.

367 **6.3.1 Ordering of Results**

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

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

371 **6.3.2 Show**

372 This section describes how to implement the *show* verb when applied to an instance of
 373 *CIM_HostedAccessPoint*. Implementations shall support the use of the *show* verb with
 374 *CIM_HostedAccessPoint*.

375 **6.3.2.1 Show Command Form for Multiple Instances – CIM_ComputerSystem Reference**

376 This command form is used when the *show* verb applies to multiple instances. This command form
 377 corresponds to the *show* command issued against instances of *CIM_HostedAccessPoint* where only one
 378 reference is specified and the reference is to an instance of *CIM_ComputerSystem*.

379 **6.3.2.1.1 Command Form**

380 `show <CIM_HostedAccessPoint multiple instances>`

381 **6.3.2.1.2 CIM Requirements**

382 See *CIM_HostedAccessPoint* in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
 383 mandatory properties.

384 **6.3.2.1.3 Behavior Requirements**

385 **6.3.2.1.3.1 Preconditions**

386 *\$instance* represents the instance of *CIM_ComputerSystem*, which is referenced by
 387 *CIM_HostedAccessPoint*.

388 6.3.2.1.3.2 Pseudo Code

```
389 &smShowAssociationInstances ( "CIM_HostedAccessPoint", $instance.GetObjectPath() );  
390 &smEnd;
```

391 6.3.2.2 Show Command Form for a Single Instance – CIM_USBRedirectionSAP Reference

392 This command form is used when the `show` verb applies to a single instance. The command form
393 corresponds to the `show` verb issued against instances of `CIM_HostedAccessPoint` where only one
394 reference is specified and the reference is to an instance of `CIM_USBRedirectionSAP`.

395 6.3.2.2.1 Command Form

```
396 show <CIM_HostedAccessPoint single instance>
```

397 6.3.2.2.2 CIM Requirements

398 See `CIM_HostedAccessPoint` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
399 mandatory properties.

400 6.3.2.2.3 Behavior Requirements**401 6.3.2.2.3.1 Preconditions**

402 `$instance` represents the instance of `CIM_USBRedirectionSAP`, which is referenced by
403 `CIM_HostedAccessPoint`.

404 6.3.2.2.3.2 Pseudo Code

```
405 &smShowAssociationInstances ( "CIM_HostedAccessPoint", $instance.GetObjectPath() );  
406 &smEnd;
```

407 6.3.2.3 Show Command Form for a Single Instance – Both References

408 This command form is used when the `show` verb applies to a single instance. This command form
409 corresponds to the `show` command issued against `CIM_HostedAccessPoint` where both references are
410 specified; therefore, the desired instance is unambiguously identified.

411 6.3.2.3.1 Command Form

```
412 show <CIM_HostedAccessPoint single instance>
```

413 6.3.2.3.2 CIM Requirements

414 See `CIM_HostedAccessPoint` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
415 mandatory properties.

416 6.3.2.3.3 Behavior Requirements**417 6.3.2.3.3.1 Preconditions**

418 `$instanceA` represents the referenced instance of `CIM_ComputerSystem` through
419 `CIM_HostedAccessPoint` association.

420 `$instanceB` represents the other instance of `CIM_USBRedirectionSAP` which is referenced by
421 `CIM_HostedAccessPoint`.

422 6.3.2.3.3.2 Pseudo Code

```
423 &smShowAssociationInstance ( "CIM_HostedAccessPoint", $instanceA.getObjectPath(),
424     $instanceB.getObjectPath() );
425 &smEnd;
```

426 6.4 CIM_ServiceAffectsElement

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

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

433 **Table 4 – 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 6.4.2.
start	Not supported	
stop	Not supported	

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

436 6.4.1 Ordering of Results

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

- 439 • Results for `CIM_ServiceAffectsElement` are unordered; therefore, no algorithm is defined.

440 6.4.2 Show

441 This section describes how to implement the `show` verb when applied to an instance of
 442 `CIM_ServiceAffectsElement`. Implementations shall support the use of the `show` verb with
 443 `CIM_ServiceAffectsElement`.

444 6.4.2.1 Show Command Form for Multiple Instances – CIM_ComputerSystem Reference

445 This command form is used when the `show` verb applies to multiple instances. This command form
 446 corresponds to the `show` command issued against instances of `CIM_ServiceAffectsElement` where only
 447 one reference is specified and the reference is to an instance of `CIM_ComputerSystem`.

448 6.4.2.1.1 Command Form

```
449 show <CIM_ServiceAffectsElement multiple instances>
```

450 6.4.2.1.2 CIM Requirements

451 See CIM_ServiceAffectsElement in the “CIM Elements” section of the [USB Redirection Profile](#) for the list
452 of mandatory properties.

453 6.4.2.1.3 Behavior Requirements

454 6.4.2.1.3.1 Preconditions

455 \$instance represents the instance of CIM_ComputerSystem, which is referenced by
456 CIM_ServiceAffectsElement.

457 6.4.2.1.3.2 Pseudo Code

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

461 6.4.2.2 Show Command Form for Multiple Instances – CIM_USBDevice Reference

462 This command form is used when the show verb applies to multiple instances. This command form
463 corresponds to the show command issued against instances of CIM_ServiceAffectsElement where only
464 one reference is specified and the reference is to an instance of CIM_USBDevice.

465 6.4.2.2.1 Command Form

```
466 show <CIM_ServiceAffectsElement multiple instances>
```

467 6.4.2.2.2 CIM Requirements

468 See CIM_ServiceAffectsElement in the “CIM Elements” section of the [USB Redirection Profile](#) for the list
469 of mandatory properties.

470 6.4.2.2.3 Behavior Requirements

471 6.4.2.2.3.1 Preconditions

472 \$instance represents the instance of CIM_USBDevice, which is referenced by
473 CIM_ServiceAffectsElement.

474 6.4.2.2.3.2 Pseudo Code

```
475 &smShowAssociationInstances ( "CIM_ServiceAffectsElement",  
476     $instance.getObjectPath() );  
477 &smEnd;
```

478 6.4.2.3 Show Command Form for Multiple Instances – CIM_USBRedirectionService Reference

479 This command form is used when the show verb applies to multiple instances. This command form
480 corresponds to the show command issued against instances of CIM_ServiceAffectsElement where only
481 one reference is specified and the reference is to an instance of CIM_USBRedirectionService.

482 6.4.2.3.1 Command Form

```
483 show <CIM_ServiceAffectsElement multiple instances>
```

484 6.4.2.3.2 CIM Requirements

485 See CIM_ServiceAffectsElement in the “CIM Elements” section of the [USB Redirection Profile](#) for the list
486 of mandatory properties.

487 6.4.2.3.3 Behavior Requirements

488 6.4.2.3.3.1 Preconditions

489 \$instance represents the instance of CIM_USBRedirectionService, which is referenced by
490 CIM_ServiceAffectsElement.

491 6.4.2.3.3.2 Pseudo Code

```
492 &smShowAssociationInstances ( "CIM_ServiceAffectsElement",
493     $instance.GetObjectPath() );
494 &smEnd;
```

495 6.4.2.4 Show Command Form for a Single Instance – Both References

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

499 6.4.2.4.1 Command Form

```
500 show <CIM_ServiceAffectsElement single instance>
```

501 6.4.2.4.2 CIM Requirements

502 See CIM_ServiceAffectsElement in the “CIM Elements” section of the [USB Redirection Profile](#) for the list
503 of mandatory properties.

504 6.4.2.4.3 Behavior Requirements

505 6.4.2.4.3.1 Preconditions

506 \$instanceA represents the instance of CIM_USBRedirectionService which is referenced by
507 CIM_ServiceAffectsElement.

508 \$instanceB represents the instance of CIM_ComputerSystem or CIM_USBDevice which is referenced
509 by CIM_ServiceAffectsElement.

510 6.4.2.4.3.2 Pseudo Code

```
511 &smShowAssociationInstance ( "CIM_ServiceAffectsElement", $instanceA.GetObjectPath(),
512     $instanceB.GetObjectPath() );
513 &smEnd;
```

514 6.5 CIM_SAPAvailableForElement

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

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

521

Table 5 – Command Verb Requirements for CIM_SAPAvailableForElement

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	

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

524 6.5.1 Ordering of Results

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

- 527 • Results for `CIM_SAPAvailableForElement` are unordered; therefore, no algorithm is defined.

528 6.5.2 Show

529 This section describes how to implement the `show` verb when applied to an instance of
 530 `CIM_SAPAvailableForElement`. Implementations shall support the use of the `show` verb with
 531 `CIM_SAPAvailableForElement`.

532 6.5.2.1 Show Command Form for Multiple Instances – `CIM_ComputerSystem` Reference

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

536 6.5.2.1.1 Command Form

537 `show <CIM_SAPAvailableToElement multiple instances>`

538 6.5.2.1.2 CIM Requirements

539 See `CIM_SAPAvailableToElement` in the “CIM Elements” section of the [USB Redirection Profile](#) for the
 540 list of mandatory properties.

541 6.5.2.1.3 Behavior Requirements

542 6.5.2.1.3.1 Preconditions

543 `$instance` represents the instance of `CIM_ComputerSystem`, which is referenced by
 544 `CIM_SAPAvailableToElement`.

545 6.5.2.1.3.2 Pseudo Code

```
546 &smShowAssociationInstances ( "CIM_SAPAvailableToElement",
547     $instance.GetObjectPath() );
548 &smEnd;
```

549 6.5.2.2 Show Command Form for a Single Instance – CIM_USBDevice Reference

550 This command form is used when the `show` verb applies to a single instance. This command form
551 corresponds to the `show` command issued against instances of `CIM_SAPAvailableToElement` where
552 only one reference is specified and the reference is to an instance of `CIM_USBDevice`.

553 6.5.2.2.1 Command Form

```
554 show <CIM_SAPAvailableToElement single instance>
```

555 6.5.2.2.2 CIM Requirements

556 See `CIM_SAPAvailableToElement` in the “CIM Elements” section of the [USB Redirection Profile](#) for the
557 list of mandatory properties.

558 6.5.2.2.3 Behavior Requirements

559 6.5.2.2.3.1 Preconditions

560 `$instance` represents the instance of `CIM_USBDevice`, which is referenced by
561 `CIM_SAPAvailableToElement`.

562 6.5.2.2.3.2 Pseudo Code

```
563 &smShowAssociationInstances ( "CIM_SAPAvailableToElement",
564     $instance.GetObjectPath() );
565 &smEnd;
```

566 6.5.2.3 Show Command Form for Multiple Instances – CIM_USBRedirectionSAP Reference

567 This command form is used when the `show` verb applies to multiple instances. This command form
568 corresponds to the `show` command issued against instances of `CIM_SAPAvailableToElement` where
569 only one reference is specified and the reference is to an instance of `CIM_USBRedirectionSAP`.

570 6.5.2.3.1 Command Form

```
571 show <CIM_SAPAvailableToElement multiple instances>
```

572 6.5.2.3.2 CIM Requirements

573 See `CIM_SAPAvailableToElement` in the “CIM Elements” section of the [USB Redirection Profile](#) for the
574 list of mandatory properties.

575 6.5.2.3.3 Behavior Requirements

576 6.5.2.3.3.1 Preconditions

577 `$instance` represents the instance of `CIM_USBRedirectionSAP`, which is referenced by
578 `CIM_SAPAvailableToElement`.

579 **6.5.2.3.2 Pseudo Code**

```
580 &smShowAssociationInstances ( "CIM_SAPAvailableToElement",
581     $instance.GetObjectPath() );
582 &smEnd;
```

583 **6.5.2.4 Show Command Form for a Single Instance – Both References**

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

587 **6.5.2.4.1 Command Form**

```
588 show <CIM_SAPAvailableForElement single instance>
```

589 **6.5.2.4.2 CIM Requirements**

590 See `CIM_SAPAvailableToElement` in the “CIM Elements” section of the [USB Redirection Profile](#) for the
 591 list of mandatory properties.

592 **6.5.2.4.3 Behavior Requirements**593 **6.5.2.4.3.1 Preconditions**

594 `$instanceA` represents the instance of `CIM_USBRedirectionSAP`, which is referenced by
 595 `CIM_SAPAvailableForElement`.

596 `$instanceB` represents the instance of `CIM_ComputerSystem` or `CIM_USBDevice` which is referenced
 597 by `CIM_SAPAvailableForElement`.

598 **6.5.2.4.3.2 Pseudo Code**

```
599 &smShowAssociationInstance ( "CIM_SAPAvailableForElement", $instanceA.GetObjectPath(),
600     $instanceB.GetObjectPath() );
601 &smEnd;
```

602 **6.6 CIM_ServiceAccessBySAP**

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

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

609 **Table 6 – Command Verb Requirements for CIM_ServiceAccessBySAP**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	

Command Verb	Requirement	Comments
set	Not supported	
show	Shall	See 6.6.2.
start	Not supported	
stop	Not supported	

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

612 6.6.1 Ordering of Results

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

- 615 • Results for `CIM_ServiceAccessBySAP` are unordered; therefore, no algorithm is defined.

616 6.6.2 Show

617 This section describes how to implement the `show` verb when applied to an instance of
618 `CIM_ServiceAccessBySAP`. Implementations shall support the use of the `show` verb with
619 `CIM_ServiceAccessBySAP`.

620 6.6.2.1 Show Command Form for a Single Instance – `CIM_USBRedirectionSAP` Reference

621 This command form is used when the `show` verb applies to a single instance. This command form
622 corresponds to the `show` command issued against instances of `CIM_ServiceAccessBySAP` where only
623 one reference is specified and the reference is to an instance of `CIM_USBRedirectionSAP`.

624 6.6.2.1.1 Command Form

```
625 show <CIM_ServiceAccessBySAP single instance>
```

626 6.6.2.1.2 CIM Requirements

627 See `CIM_ServiceAccessBySAP` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list
628 of mandatory properties.

629 6.6.2.1.3 Behavior Requirements

630 6.6.2.1.3.1 Preconditions

631 `$instance` represents the instance of `CIM_USBRedirectionSAP`, which is referenced by
632 `CIM_ServiceAccessBySAP`.

633 6.6.2.1.3.2 Pseudo Code

```
634 &smShowAssociationInstances ( "CIM_ServiceAccessBySAP", $instance.GetObjectPath() );  
635 &smEnd;
```

636 6.6.2.2 Show Command Form for Multiple Instances – `CIM_USBRedirectionService` Reference

637 This command form is used when the `show` verb applies to multiple instances. The command form
638 corresponds to the `show` verb issued against instances of `CIM_ServiceAccessBySAP` where only one
639 reference is specified and the reference is to an instance of `CIM_USBRedirectionService`.

640 6.6.2.2.1 Command Form

```
641 show <CIM_ServiceAccessBySAP multiple instances>
```

642 6.6.2.2.2 CIM Requirements

643 See CIM_ServiceAccessBySAP in the “CIM Elements” section of the [USB Redirection Profile](#) for the list
644 of mandatory properties.

645 6.6.2.2.3 Behavior Requirements

646 6.6.2.2.3.1 Preconditions

647 \$instance represents the instance of CIM_USBRedirectionService, which is referenced by
648 CIM_ServiceAccessBySAP.

649 6.6.2.2.3.2 Pseudo Code

```
650 &smShowAssociationInstances ( "CIM_ServiceAccessBySAP", $instance.getObjectPath() );  
651 &smEnd;
```

652 6.6.2.3 Show Command Form for a Single Instance – Both References

653 This command form is used when the show verb applies to a single instance. This command form
654 corresponds to the show command issued against CIM_ServiceAccessBySAP where both references are
655 specified; and therefore, the desired instance is unambiguously identified.

656 6.6.2.3.1 Command Form

```
657 show <CIM_ServiceAccessBySAP single instance>
```

658 6.6.2.3.2 CIM Requirements

659 See CIM_ServiceAccessBySAP in the “CIM Elements” section of the [USB Redirection Profile](#) for the list
660 of mandatory properties.

661 6.6.2.3.3 Behavior Requirements

662 6.6.2.3.3.1 Preconditions

663 \$instanceA represents the referenced instance of CIM_USBRedirectionService through
664 CIM_ServiceAccessBySAP association.

665 \$instanceB represents the other instance of CIM_USBRedirectionSAP which is referenced by
666 CIM_ServiceAccessBySAP.

667 6.6.2.3.3.2 Pseudo Code

```
668 &smShowAssociationInstance ( "CIM_ServiceAccessBySAP", $instanceA.getObjectPath(),  
669     $instanceB.getObjectPath() );  
670 &smEnd;
```

671 6.7 CIM_RemoteAccessAvailableToElement

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

673 Table 7 lists each SM CLP verb, the required level of support for the verb in conjunction with the target
674 class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and
675 target. Table 7 is for informational purposes only; in case of a conflict between Table 7 and requirements

676 detailed in the following sections, the text detailed in the following sections supersedes the information in
 677 Table 7.

678 **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	

679 No mappings are defined for the following verbs for the specified target: *create*, *delete*, *dump*, *load*,
 680 *reset*, *set*, *start*, and *stop*.

681 **6.7.1 Ordering of Results**

682 When results are returned for multiple instances of *CIM_RemoteAccessAvailableToElement*,
 683 implementations shall utilize the following algorithm to produce the natural (that is, default) ordering:

- 684 • Results for *CIM_RemoteAccessAvailableToElement* are unordered; therefore, no algorithm is
 685 defined.

686 **6.7.2 Show**

687 This section describes how to implement the *show* verb when applied to an instance of
 688 *CIM_RemoteAccessAvailableToElement*. Implementations shall support the use of the *show* verb with
 689 *CIM_RemoteAccessAvailableToElement*.

690 **6.7.2.1 Show Command Form for a Single Instance – CIM_RemoteServiceAccessPoint Reference**

691 This command form is used when the *show* verb applies to a single instance. This command form
 692 corresponds to the *show* command issued against instances of *CIM_RemoteAccessAvailableToElement*
 693 where only one reference is specified and the reference is to an instance of
 694 *CIM_RemoteServiceAccessPoint*.

695 **6.7.2.1.1 Command Form**

```
696 show <CIM_RemoteAccessAvailableToElement single instance>
```

697 **6.7.2.1.2 CIM Requirements**

698 See *CIM_RemoteAccessAvailableToElement* in the “CIM Elements” section of the [USB Redirection](#)
 699 [Profile](#) for the list of mandatory properties.

700 6.7.2.1.3 Behavior Requirements

701 6.7.2.1.3.1 Preconditions

702 \$instance represents the instance of CIM_RemoteServiceAccessPoint, which is referenced by
703 CIM_RemoteAccessAvailableToElement.

704 6.7.2.1.3.2 Pseudo Code

```
705 &smShowAssociationInstances ( "CIM_RemoteAccessAvailableToElement",  
706     $instance.GetObjectPath() );  
707 &smEnd;
```

708 6.7.2.2 Show Command Form for a Single Instance – CIM_USBRedirectionSAP Reference

709 This command form is used when the `show` verb applies to a single instance. The command form
710 corresponds to the `show` verb issued against instances of CIM_RemoteAccessAvailableToElement where
711 only one reference is specified and the reference is to an instance of CIM_USBRedirectionSAP.

712 6.7.2.2.1 Command Form

```
713 show <CIM_RemoteAccessAvailableToElement single instance>
```

714 6.7.2.2.2 CIM Requirements

715 See CIM_RemoteAccessAvailableToElement in the “CIM Elements” section of the [USB Redirection](#)
716 [Profile](#) for the list of mandatory properties.

717 6.7.2.2.3 Behavior Requirements

718 6.7.2.2.3.1 Preconditions

719 \$instance represents the instance of CIM_USBRedirectionSAP, which is referenced by
720 CIM_RemoteAccessAvailableToElement.

721 6.7.2.2.3.2 Pseudo Code

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

725 6.7.2.3 Show Command Form for a Single Instance – Both References

726 This command form is used when the `show` verb applies to a single instance. This command form
727 corresponds to the `show` command issued against CIM_RemoteAccessAvailableToElement where both
728 references are specified; therefore, the desired instance is unambiguously identified.

729 6.7.2.3.1 Command Form

```
730 show <CIM_RemoteAccessAvailableToElement single instance>
```

731 6.7.2.3.2 CIM Requirements

732 See CIM_RemoteAccessAvailableToElement in the “CIM Elements” section of the [USB Redirection](#)
733 [Profile](#) for the list of mandatory properties.

734 **6.7.2.3.3 Behavior Requirements**

735 **6.7.2.3.3.1 Preconditions**

736 \$instanceA represents the referenced instance of CIM_RemoteServiceAccessPoint through
 737 CIM_RemoteAccessAvailableToElement association.

738 \$instanceB represents the other instance of CIM_USBRedirectionSAP which is referenced by
 739 CIM_RemoteAccessAvailableToElement.

740 **6.7.2.3.3.2 Pseudo Code**

```
741 &smShowAssociationInstance ( "CIM_RemoteAccessAvailableToElement",
742     $instanceA.getObjectPath(), $instanceB.getObjectPath() );
743 &smEnd;
```

744 **6.8 CIM_BindsTo**

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

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

751 **Table 8 – 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 6.8.2.
start	Not supported	
stop	Not supported	

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

754 **6.8.1 Ordering of Results**

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

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

758 6.8.2 Show

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

761 6.8.2.1 Show Command Form for Multiple Instances – `CIM_ProtocolEndpoint` Reference

762 This command form is used when the `show` verb applies to multiple instances. This command form
763 corresponds to the `show` command issued against instances of `CIM_BindsTo` where only one reference
764 is specified and the reference is to an instance of `CIM_ProtocolEndpoint`.

765 6.8.2.1.1 Command Form

```
766 show <CIM_BindsTo multiple instances>
```

767 6.8.2.1.2 CIM Requirements

768 See `CIM_BindsTo` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of mandatory
769 properties.

770 6.8.2.1.3 Behavior Requirements

771 6.8.2.1.3.1 Preconditions

772 `$instance` represents the instance of `CIM_ProtocolEndpoint`, which is referenced by `CIM_BindsTo`.

773 6.8.2.1.3.2 Pseudo Code

```
774 &smShowAssociationInstances ( "CIM_BindsTo", $instance.GetObjectPath() );  
775 &smEnd;
```

776 6.8.2.2 Show Command Form for a Single Instance – `CIM_USBRedirectionSAP` Reference

777 This command form is used when the `show` verb applies to a single instance. This command form
778 corresponds to the `show` command issued against instances of `CIM_BindsTo` where only one reference
779 is specified and the reference is to an instance of `CIM_USBRedirectionSAP`.

780 6.8.2.2.1 Command Form

```
781 show <CIM_BindsTo single instance>
```

782 6.8.2.2.2 CIM Requirements

783 See `CIM_BindsTo` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of mandatory
784 properties.

785 6.8.2.2.3 Behavior Requirements

786 6.8.2.2.3.1 Preconditions

787 `$instance` represents the instance of `CIM_USBRedirectionSAP`, which is referenced by `CIM_BindsTo`.

788 6.8.2.2.3.2 Pseudo Code

```
789 &smShowAssociationInstances ( "CIM_BindsTo", $instance.GetObjectPath() );  
790 &smEnd;
```

791 **6.8.2.3 Show Command Form for a Single Instance – Both References**

792 This command form is used when the `show` verb applies to a single instance. This command form
 793 corresponds to the `show` command issued against `CIM_BindsTo` where both references are specified;
 794 therefore, the desired instance is unambiguously identified.

795 **6.8.2.3.1 Command Form**

796 `show <CIM_BindsTo single instance>`

797 **6.8.2.3.2 CIM Requirements**

798 See `CIM_BindsTo` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of mandatory
 799 properties.

800 **6.8.2.3.3 Behavior Requirements**

801 **6.8.2.3.3.1 Preconditions**

802 `$instanceA` represents the referenced instance of `CIM_ProtocolEndpoint` through `CIM_BindsTo`
 803 association.

804 `$instanceB` represents the other instance of `CIM_USBRedirectionSAP` which is referenced by
 805 `CIM_BindsTo`.

806 **6.8.2.3.3.2 Pseudo Code**

807 `&smShowAssociationInstance ("CIM_BindsTo", $instanceA.getObjectPath(),`
 808 `$instanceB.getObjectPath());`
 809 `&smEnd;`

810 **6.9 CIM_LogicalIdentity**

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

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

817 **Table 9 – Command Verb Requirements for CIM_LogicalIdentity**

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	

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

820 6.9.1 Ordering of Results

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

- 823 • Results for CIM_LogicalIdentity are unordered; therefore, no algorithm is defined.

824 6.9.2 Show

825 This section describes how to implement the `show` verb when applied to an instance of
826 CIM_LogicalIdentity. Implementations shall support the use of the `show` verb with CIM_LogicalIdentity.

827 6.9.2.1 Show Command Form for a Single Instance – CIM_USBDevice Reference

828 This command form is used when the `show` verb applies to a single instance. The command form
829 corresponds to the `show` verb issued against instances of CIM_LogicalIdentity where only one reference
830 is specified and the reference is to an instance of CIM_USBDevice.

831 6.9.2.1.1 Command Form

```
832 show <CIM_LogicalIdentity single instance>
```

833 6.9.2.1.2 CIM Requirements

834 See CIM_LogicalIdentity in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
835 mandatory properties.

836 6.9.2.1.3 Behavior Requirements

837 6.9.2.1.3.1 Preconditions

838 `$instance` represents the instance of CIM_USBDevice, which is referenced by CIM_LogicalIdentity.

839 6.9.2.1.3.2 Pseudo Code

```
840 &smShowAssociationInstances ( "CIM_LogicalIdentity", $instance.GetObjectPath() );  
841 &smEnd;
```

842 6.9.2.2 Show Command Form for a Single Instance – A Concrete Subclass of CIM_LogicalDevice 843 Reference

844 This command form is used when the `show` verb applies to a single instance. The command form
845 corresponds to the `show` verb issued against instances of CIM_LogicalIdentity where only one reference
846 is specified and the reference is to an instance of a concrete subclass of CIM_LogicalDevice.

847 6.9.2.2.1 Command Form

```
848 show <CIM_LogicalIdentity single instance>
```

849 6.9.2.2.2 CIM Requirements

850 See CIM_LogicalIdentity in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
851 mandatory properties.

852 6.9.2.2.3 Behavior Requirements

853 6.9.2.2.3.1 Preconditions

854 `$instance` represents the instance of a concrete subclass of `CIM_LogicalDevice`, which is referenced
855 by `CIM_LogicalIdentity`.

856 6.9.2.2.3.2 Pseudo Code

```
857 &smShowAssociationInstances ( "CIM_LogicalIdentity", $instance.GetObjectPath() );
858 &smEnd;
```

859 6.9.2.3 Show Command Form for a Single Instance – Both References

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

863 6.9.2.3.1 Command Form

```
864 show <CIM_LogicalIdentity single instance>
```

865 6.9.2.3.2 CIM Requirements

866 See `CIM_LogicalIdentity` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
867 mandatory properties.

868 6.9.2.3.3 Behavior Requirements

869 6.9.2.3.3.1 Preconditions

870 `$instanceA` represents the instance of `CIM_USBDevice` which is referenced by `CIM_LogicalIdentity`.

871 `$instanceB` represents the instance of a concrete subclass of `CIM_LogicalDevice` which is referenced
872 by `CIM_LogicalIdentity`.

873 6.9.2.3.3.2 Pseudo Code

```
874 &smShowAssociationInstance ( "CIM_LogicalIdentity", $instanceA.GetObjectPath(),
875     $instanceB.GetObjectPath() );
876 &smEnd;
```

877 6.10 CIM_USBDevice

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

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

884

Table 10 – Command Verb Requirements for CIM_USBDevice

Command Verb	Requirement	Comments
create	May	See 6.10.2.
delete	May	See 6.10.3.
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.10.4.
start	Not supported	
stop	Not supported	

885 No mappings are defined for the following verbs for the specified target: dump, load, reset, set,
886 start, and stop.

887 6.10.1 Ordering of Results

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

- 890 • Results for CIM_USBDevice are unordered; therefore, no algorithm is defined.

891 6.10.2 Create

892 This section describes how to implement the `create` verb when applied to an instance of CIM_Account.
893 Implementations may support the use of the `create` verb with CIM_USBDevice.

894 The `create` verb is used to create a USB Device on a system.

895 6.10.2.1 Create Specifying Properties

896 6.10.2.1.1 Command Form

```
897 create <CIM_USBDevice> <propertyname1>=<propertyvalue1>  
898 <propertynamen>=<propertyvaluen>
```

899 6.10.2.1.2 CIM Requirements

900 See CIM_USBRedirectionService.CreateUSBDevice in the “CIM Elements” section of the [USB](#)
901 [Redirection Profile](#) for the list of mandatory properties.

902 6.10.2.1.3 Behavior Requirements

903 6.10.2.1.3.1 Preconditions

904 `$system` contains the instance of CIM_ComputerSystem that is the Resultant Target of the CLP
905 command. This is the container instance for the desired Account instance.

906 **6.10.2.1.3.2 Pseudo Code**

```

907 //1 Find the Service for the system to create the USB Device
908 #Error = &smOpAssociators(
909     $system.getObjectPath(),
910     "CIM_HostedService",
911     "CIM_USBRedirectionService",
912     NULL,
913     NULL,
914     NULL,
915     $SvcInstancePaths[])
916 if (0 != #Error.code)
917     {
918     &smProcessOpError (#Error);
919     //includes &smEnd;
920     }
921 $service = $SvcInstancePaths[0];
922 //2 populate the properties
923 $USBDeviceTemplate = smNewInstance ("CIM_USBDevice");
924 for #i < n
925     {
926     $USBDeviceTemplate.<propertname#i> = <propertyvalue#i>;
927     }
928 //3 invoke the method
929 $USBDevice = smNewInstance ("CIM_USBDevice");
930 //build the parameter lists and invoke the method
931 %InArguments[] = {newArgument("NewUSBDevice", $USBDeviceTemplate)}
932 %OutArguments[] = { newArgument("USBDevice",
933     $USBDevice.GetObjectPath()) };
934 //invoke method
935 #returnStatus = smOpInvokeMethod ($service.GetObjectPath(),
936     "CreateUSBDevice",
937     %InArguments[],
938     %OutArguments[]);
939 //3 process return code to CLP Command Status
940 if (0 != #Error.code)
941     {
942     //method invocation failed
943     if ( (null != #Error.$error) && (null != #Error.$error[0]) )
944         {
945         // if the method invocation contains an embedded error
946         // use it for the Error for the overall job
947         &smAddError($job, #Error.$error[0]);
948         &smMakeCommandStatus($job);
949         &smEnd;
950         }
951     else if (#Error.code == 17)
952         {
953         //trap for CIM_METHOD_NOT_FOUND
954         //and make nice Unsupported msg.

```

```
955     //unsupported
956     $OperationError = smNewInstance("CIM_Error");
957     //CIM_ERR_NOT_SUPPORTED
958     $OperationError.CIMStatusCode = 7;
959     //Other
960     $OperationError.ErrorType = 1;
961     //Low
962     $OperationError.PerceivedSeverity = 2;
963     $OperationError.OwningEntity = DMTF:SMCLP;
964     $OperationError.MessageID = 0x00000001;
965     $OperationError.Message = "Operation is not supported.";
966     &smAddError($job, $OperationError);
967     &smMakeCommandStatus($job);
968     &smEnd;
969 }
970 else
971 {
972     //operation failed, but no detailed error instance, need to make one up
973     //make an Error instance and associate with job for Operation
974     $OperationError = smNewInstance("CIM_Error");
975     //CIM_ERR_FAILED
976     $OperationError.CIMStatusCode = 1;
977     //Software Error
978     $OperationError.ErrorType = 4;
979     //Unknown
980     $OperationError.PerceivedSeverity = 0;
981     $OperationError.OwningEntity = DMTF:SMCLP;
982     $OperationError.MessageID = 0x00000009;
983     $OperationError.Message = "An internal software error has occurred.";
984     &smAddError($job, $OperationError);
985     &smMakeCommandStatus($job);
986     &smEnd;
987 }
988 }
989 //if CIM op failed
990 else if (0 == #returnStatus)
991 {
992     //completed successfully, show created instance
993     &smShowInstance($USBDevice.getObjectPath(), NULL);
994     &smEnd;
995 }
996 else if (4 == #returnStatus)
997 {
998     //generic failure
999     $OperationError = smNewInstance("CIM_Error");
1000     //CIM_ERR_FAILED
1001     $OperationError.CIMStatusCode = 1;
1002     //Other
1003     $OperationError.ErrorType = 1;
```

```

1004 //Low
1005 $OperationError.PerceivedSeverity = 2;
1006 $OperationError.OwningEntity = DMTF:SMCLP;
1007 $OperationError.MessageID = 0x00000002;
1008 $OperationError.Message = "Failed. No further information is available.";
1009 &smAddError($job, $OperationError);
1010 &smMakeCommandStatus($job);
1011 }
1012 else
1013 {
1014 //invalid parameter
1015 $OperationError = smNewInstance("CIM_Error");
1016 //CIM_ERR_FAILED
1017 $OperationError.CIMStatusCode = 1;
1018 //Other
1019 $OperationError.ErrorType = 1;
1020 //Low
1021 $OperationError.PerceivedSeverity = 2;
1022 $OperationError.OwningEntity = DMTF:SMCLP;
1023 $OperationError.MessageID = 0x00000004;
1024 $OperationError.Message = "One or more parameters specified are invalid.";
1025 &smAddError($job, $OperationError);
1026 &smMakeCommandStatus($job);
1027 &smEnd;
1028 }

```

1029 6.10.3 Delete

1030 This section describes how to implement the `delete` verb when applied to an instance of
 1031 `CIM_USBDevice`. Implementations may support the use of the `delete` verb with `CIM_USBDevice`.

1032 The `delete` command is used to remove an instance of `CIM_USBDevice`.

1033 6.10.3.1 Delete a Single Instance

1034 This command is used to delete a single instance of `CIM_USBDevice`.

1035 6.10.3.1.1 Command Form

```
1036 delete <CIM_USBDevice single instance>
```

1037 6.10.3.1.2 CIM Requirements

1038 See `CIM_USBDevice` in the "CIM Elements" section of the [USB Redirection Profile](#) for the list of
 1039 mandatory properties.

1040 6.10.3.1.3 Behavior Requirements

```

1041 $instance=<CIM_USBDevice single instance>
1042 &smOpDeleteInstance ( $instance.GetObjectPath() );
1043 &smEnd;

```

1044 6.10.4 Show

1045 The `show` verb is used to display information about instances of `CIM_USBDevice`. Implementations shall
1046 support the use of the `show` verb with `CIM_USBDevice`.

1047 6.10.4.1 Show a Single Instance

1048 This command form is used to display the information about a single instance of `CIM_USBDevice`.

1049 6.10.4.1.1 Command Form

```
1050 show <CIM_USBDevice single instance>
```

1051 6.10.4.1.2 CIM Requirements

1052 See `CIM_USBDevice` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
1053 mandatory properties.

1054 6.10.4.1.3 Behavior Requirements

1055 6.10.4.1.3.1 Preconditions

1056 `$instance` represents the instance of `CIM_USBDevice`.

1057 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

1058 `#propertylist[]` is an array of mandatory non-key property names.

1059 6.10.4.1.3.2 Pseudo Code

```
1060 if (false != #all) { #propertylist[] = NULL; }  
1061 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );  
1062 &smEnd;
```

1063 6.10.4.2 Show Multiple Instances – `CIM_ServiceAvailableForElement`

1064 This command form is used to display the information about multiple instances of
1065 `CIM_RemoteServiceAccessPoint`. This command form corresponds to UFsT-based selection within a
1066 scoping system.

1067 6.10.4.2.1 Command Form

```
1068 show <CIM_USBDevice multiple instances>
```

1069 6.10.4.2.2 CIM Requirements

1070 See `CIM_USBDevice` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
1071 mandatory properties.

1072 6.10.4.2.3 Behavior Requirements

1073 6.10.4.2.3.1 Preconditions

1074 `$containerInstance` represents the instance of `CIM_USBRedirectionSAP` for which the scoped
1075 instance of `CIM_USBDevice` is being displayed. The `CIM_USBRedirectionSAP` is associated to targeted
1076 instances of `CIM_USBDevice` via a `CIM_ServiceAvailableForElement` association.

1077 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

1078 `#propertylist[]` is an array of mandatory non-key property names.

1079 **6.10.4.2.3.2 Pseudo Code**

```
1080 if (false != #all) { #propertylist[] = NULL; }
1081 &smShowInstances ( "CIM_USBDevice", "CIM_ServiceAvailableForElement",
1082     $containerInstance.getObjectPath(), #propertylist[] );
1083 &smEnd;
```

1084 **6.10.4.3 Show Multiple Instances – CIM_ServiceAffectsElement**

1085 This command form is used to display the information about multiple instances of
1086 CIM_RemoteServiceAccessPoint. This command form corresponds to UFsT-based selection within a
1087 scoping system.

1088 **6.10.4.3.1 Command Form**

```
1089 show <CIM_USBDevice multiple instances>
```

1090 **6.10.4.3.2 CIM Requirements**

1091 See CIM_USBDevice in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
1092 mandatory properties.

1093 **6.10.4.3.3 Behavior Requirements**

1094 **6.10.4.3.3.1 Preconditions**

1095 \$containerInstance represents the instance of CIM_USBRedirectionService for which the scoped
1096 instance of CIM_USBDevice is being displayed. The CIM_USBRedirectionService is associated to
1097 targeted instances of CIM_USBDevice via a CIM_ServiceAvailableForElement association.

1098 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

1099 #propertylist[] is an array of mandatory non-key property names.

1100 **6.10.4.3.3.2 Pseudo Code**

```
1101 if (false != #all) { #propertylist[] = NULL; }
1102 &smShowInstances ( "CIM_USBDevice", "CIM_ServiceAffectsElement",
1103     $containerInstance.getObjectPath(), #propertylist[] );
1104 &smEnd;
```

1105 **6.11 CIM_RemoteServiceAccessPoint**

1106 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

1107 Table 11 lists each SM CLP verb, the required level of support for the verb in conjunction with the target
1108 class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and
1109 target. Table 11 is for informational purposes only; in case of a conflict between Table 11 and
1110 requirements detailed in the following sections, the text detailed in the following sections supersedes the
1111 information in Table 11.

1112 **Table 11 – 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 6.11.2.
show	Shall	See 6.11.3.
start	Not supported	
stop	Not supported	

1113 No mappings are defined for the following verbs for the specified target: create, delete, dump, exit,
1114 load, reset, set, start, and stop.

1115 6.11.1 Ordering of Results

1116 When results are returned for multiple instances of CIM_RemoteServiceAccessPoint, implementations
1117 shall utilize the following algorithm to produce the natural (that is, default) ordering:

- 1118 • Results for CIM_RemoteServiceAccessPoint are unordered; therefore, no algorithm is defined.

1119 6.11.2 Set

1120 6.11.2.1 General Usage of Set for a Single Property

1121 This command form corresponds to the general usage of the `set` verb to modify a single property of a
1122 target instance. The setting of a single property may be deterministic.

1123 The requirements for supporting modification of a property using this command form shall be equivalent
1124 to the requirement for supporting modification of the property using the ModifyInstance operation as
1125 defined in the [USB Redirection Profile](#).

1126 6.11.2.1.1 Command Form

```
1127 set <CIM_RemoteServiceAccessPoint single object> <propertyname>=<propertyvalue>
```

1128 6.11.2.1.2 CIM Requirements

1129 See CIM_RemoteServiceAccessPoint in the “CIM Elements” section of the [USB Redirection Profile](#) for
1130 the list of mandatory properties.

1131 6.11.2.1.3 Behavior Requirements

1132 6.11.2.1.3.1 Pseudo Code

```
1133 $instance=<CIM_RemoteServiceAccessPoint single object>
1134 #propertyName[] = <propertyname>
1135 #propertyValues[] = <propertyvalue>
1136 &smSetInstance ( $instance, #propertyName, #propertyValues );
1137 &smEnd;
```

1138 6.11.2.2 General Usage of Set for Multiple Properties

1139 This command form corresponds to the general usage of the `set` verb to modify multiple properties of a
1140 target instance. The setting of multiple properties may be deterministic.

1141 The requirements for supporting modification of a property using this command form shall be equivalent
1142 to the requirement for supporting modification of the property using the `ModifyInstance` operation as
1143 defined in the [USB Redirection Profile](#).

1144 6.11.2.2.1 Command Form

```
1145 set <CIM_RemoteServiceAccessPoint multiple objects> <propertyname1>=<propertyvalue1>  
1146 <propertynameN>=<propertyvalueN>
```

1147 6.11.2.2.2 CIM Requirements

1148 See `CIM_RemoteServiceAccessPoint` in the “CIM Elements” section of the [USB Redirection Profile](#) for
1149 the list of mandatory properties.

1150 6.11.2.2.3 Behavior Requirements

1151 6.11.2.2.3.1 Preconditions

1152 `$instance` represents the instance of `CIM_RemoteServiceAccessPoint`.

1153 6.11.2.2.3.2 Pseudo Code

```
1154 for #i < n  
1155 {  
1156   #propertyName[#i] = <propertyname#i>  
1157   #propertyValues[#i] = <propertyvalue#i>  
1158 }  
1159 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );  
1160 &smEnd;
```

1161 6.11.3 Show

1162 The `show` verb is used to display information about instances of `CIM_RemoteServiceAccessPoint`.
1163 Implementations shall support the use of the `show` verb with `CIM_RemoteServiceAccessPoint`.

1164 6.11.3.1 Show a Single Instance

1165 This command form is used to display the information about a single instance of
1166 `CIM_RemoteServiceAccessPoint`.

1167 6.11.3.1.1 Command Form

```
1168 show <CIM_RemoteServiceAccessPoint single instance>
```

1169 6.11.3.1.2 CIM Requirements

1170 See `CIM_RemoteServiceAccessPoint` in the “CIM Elements” section of the [USB Redirection Profile](#) for
1171 the list of mandatory properties.

1172 **6.11.3.1.3 Behavior Requirements**1173 **6.11.3.1.3.1 Preconditions**

1174 `$instance` represents the instance of `CIM_RemoteServiceAccessPoint`.

1175 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

1176 `#propertylist[]` is an array of mandatory non-key property names.

1177 **6.11.3.1.3.2 Pseudo Code**

```
1178 if (false != #all) { #propertylist[] = NULL; }
1179 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
1180 &smEnd;
```

1181 **6.11.3.2 Show Multiple Instances – CIM_RemoteAccessAvailableToElement**

1182 This command form is used to display the information about multiple instances of
1183 `CIM_RemoteServiceAccessPoint`. This command form corresponds to UFT-based selection within a
1184 scoping system.

1185 **6.11.3.2.1 Command Form**

```
1186 show <CIM_RemoteServiceAccessPoint multiple instances>
```

1187 **6.11.3.2.2 CIM Requirements**

1188 See `CIM_RemoteServiceAccessPoint` in the “CIM Elements” section of the [USB Redirection Profile](#) for
1189 the list of mandatory properties.

1190 **6.11.3.2.3 Behavior Requirements**1191 **6.11.3.2.3.1 Preconditions**

1192 `$containerInstance` represents the instance of `CIM_USBRedirectionSAP` for which the scoped
1193 instance of `CIM_RemoteServiceAccessPoint` is being displayed. The `CIM_USBRedirectionSAP` is
1194 associated to targeted instances of `CIM_RemoteServiceAccessPoint` via a
1195 `CIM_RemoteAccessAvailableToElement` association.

1196 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

1197 `#propertylist[]` is an array of mandatory non-key property names.

1198 **6.11.3.2.3.2 Pseudo Code**

```
1199 if (false != #all) { #propertylist[] = NULL; }
1200 &smShowInstances ( "CIM_RemoteServiceAccessPoint",
1201     "CIM_RemoteAccessAvailableToElement", $containerInstance.getObjectPath(),
1202     #propertylist[] );
1203 &smEnd;
```

1204 **6.12 CIM_EnabledLogicalElementCapabilities**

1205 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

1206 Table 12 lists each SM CLP verb, the required level of support for the verb in conjunction with the target
1207 class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and
1208 target. Table 12 is for informational purposes only; in case of a conflict between Table 12 and

1209 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 1210 information in Table 12.

1211 **Table 12 – Command Verb Requirements for CIM_EnabledLogicalElementCapabilities**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.12.2.
start	Not supported	
stop	Not supported	

1212 No mappings are defined for the following verbs for the specified target: *create*, *delete*, *dump*, *load*,
 1213 *reset*, *set*, *start*, and *stop*.

1214 **6.12.1 Ordering of Results**

1215 When results are returned for multiple instances of *CIM_EnabledLogicalElementCapabilities*,
 1216 implementations shall utilize the following algorithm to produce the natural (that is, default) ordering:

- 1217 • Results for *CIM_EnabledLogicalElementCapabilities* are unordered; therefore, no algorithm is
 1218 defined.

1219 **6.12.2 Show**

1220 The *show* verb is used to display information about instances of
 1221 *CIM_EnabledLogicalElementCapabilities*. Implementations shall support the use of the *show* verb with
 1222 *CIM_EnabledLogicalElementCapabilities*.

1223 **6.12.2.1 Show a Single Instance**

1224 This command form is used to display the information about a single instance of
 1225 *CIM_EnabledLogicalElementCapabilities*.

1226 **6.12.2.1.1 Command Form**

1227 `show <CIM_EnabledLogicalElementCapabilities single instance>`

1228 **6.12.2.1.2 CIM Requirements**

1229 See *CIM_EnabledLogicalElementCapabilities* in the “CIM Elements” section of the [USB Redirection](#)
 1230 [Profile](#) for the list of mandatory properties.

1231 6.12.2.1.3 Behavior Requirements

1232 6.12.2.1.3.1 Preconditions

1233 \$instance represents the instance of CIM_EnabledLogicalElementCapabilities.

1234 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

1235 #propertylist[] is an array of mandatory non-key property names.

1236 6.12.2.1.3.2 Pseudo Code

```
1237 if (false != #all) { #propertylist[] = NULL; }
1238 &smShowInstance ( $instance.GetObjectPath(), #propertylist[] );
1239 &smEnd;
```

1240 6.12.2.2 Show Multiple Instances

1241 This command form is used to display the information about multiple instances of
1242 CIM_EnabledLogicalElementCapabilities. This command form corresponds to UFST-based selection
1243 within a scoping system.

1244 6.12.2.2.1 Command Form

```
1245 show <CIM_EnabledLogicalElementCapabilities multiple instances>
```

1246 6.12.2.2.2 CIM Requirements

1247 See CIM_EnabledLogicalElementCapabilities in the “CIM Elements” section of the [USB Redirection Profile](#)
1248 for the list of mandatory properties.

1249 6.12.2.2.3 Behavior Requirements

1250 6.12.2.2.3.1 Preconditions

1251 \$containerInstance represents the instance of CIM_ConcreteCollection with ElementName property
1252 that contains “Capabilities” and is associated to the targeted instances of
1253 CIM_EnabledLogicalElementCapabilities through the CIM_MemberOfCollection association.

1254 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

1255 #propertylist[] is an array of mandatory non-key property names.

1256 6.12.2.2.3.2 Pseudo Code

```
1257 if (false != #all) { #propertylist[] = NULL; }
1258 &smShowInstances ( "CIM_EnabledLogicalElementCapabilities", "CIM_ConcreteCollection",
1259   $containerInstance.GetObjectPath(), #propertylist[] );
1260 &smEnd;
```

1261 6.13 CIM_USBRedirectionCapabilities

1262 The cd and help verbs shall be supported as described in [DSP0216](#).

1263 Table 13 lists each SM CLP verb, the required level of support for the verb in conjunction with instances
1264 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
1265 verb and target. Table 13 is for informational purposes only; in case of a conflict between Table 13 and
1266 requirements detailed in the following sections, the text detailed in the following sections supersedes the
1267 information in Table 13.

1268

Table 13 – Command Verb Requirements for CIM_USBRedirectionCapabilities

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	

1269 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 1270 `reset`, `set`, `start`, and `stop`.

1271 6.13.1 Ordering of Results

1272 When results are returned for multiple instances of `CIM_USBRedirectionCapabilities`, implementations
 1273 shall utilize the following algorithm to produce the natural (that is, default) ordering:

- 1274 • Results for `CIM_USBRedirectionCapabilities` are unordered; therefore, no algorithm is defined.

1275 6.13.2 Show

1276 This section describes how to implement the `show` verb when applied to an instance of
 1277 `CIM_USBRedirectionCapabilities`. Implementations shall support the use of the `show` verb with
 1278 `CIM_USBRedirectionCapabilities`.

1279 The `show` verb is used to display information about an instance or instances of the
 1280 `CIM_USBRedirectionCapabilities` class.

1281 6.13.2.1 Show a Single Instance

1282 This command form is for the `show` verb applied to a single instance of `CIM_USBRedirectionCapabilities`.

1283 6.13.2.1.1 Command Form

```
1284 show <CIM_USBRedirectionCapabilities single instance>
```

1285 6.13.2.1.2 CIM Requirements

1286 See `CIM_USBRedirectionCapabilities` in the “CIM Elements” section of the [USB Redirection Profile](#) for the
 1287 list of mandatory properties.

1288 6.13.2.1.3 Behavior Requirements

1289 6.13.2.1.3.1 Preconditions

1290 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

1291 **6.13.2.1.3.2 Pseudo Code**

```

1292 $instance=<CIM_USBRedirectionCapabilities single instance>
1293 #propertylist[] = NULL;
1294 if ( false == #all)
1295     {
1296         #propertylist[] = {/all mandatory non-key properties}
1297     }
1298 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
1299 &smEnd;

```

1300 **6.13.2.2 Show Multiple Instances**

1301 This command form is for the `show` verb applied to multiple instances of
 1302 `CIM_USBRedirectionCapabilities`. This command form corresponds to UFsT-based selection within a
 1303 capabilities collection.

1304 **6.13.2.2.1 Command Form**

```

1305 show <CIM_USBRedirectionCapabilities multiple instances>

```

1306 **6.13.2.2.2 CIM Requirements**

1307 See `CIM_USBRedirectionCapabilities` in the “CIM Elements” section of the [USB Redirection Profile](#) for the
 1308 list of mandatory properties.

1309 **6.13.2.2.3 Behavior Requirements**1310 **6.13.2.2.3.1 Preconditions**

1311 `$containerInstance` represents the instance of `CIM_ConcreteCollection` with `ElementName` property
 1312 that contains “Capabilities” and is associated to the targeted instances of
 1313 `CIM_USBRedirectionCapabilities` through the `CIM_MemberOfCollection` association.

1314 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

1315 **6.13.2.2.3.2 Pseudo Code**

```

1316 #propertylist[] = NULL;
1317 if ( false == #all)
1318     {
1319         #propertylist[] = {/all mandatory non-key properties }
1320     }
1321 &smShowInstances ( "CIM_USBRedirectionCapabilities", "CIM_MemberOfCollection",
1322     $containerInstance.getObjectPath(), #propertylist[] );
1323 &smEnd;

```

1324 **6.14 CIM_USBRedirectionSAP**

1325 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

1326 Table 14 lists each SM CLP verb, the required level of support for the verb in conjunction with the target
 1327 class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and
 1328 target. Table 14 is for informational purposes only; in case of a conflict between Table 14 and
 1329 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 1330 information in Table 14.

1331

Table 14 – Command Verb Requirements for CIM_USBRedirectionSAP

Command Verb	Requirement	Comments
create	May	See 6.14.2.
delete	May	See 6.14.3.
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.14.4.
show	Shall	See 6.14.5.
start	May	See 6.14.6.
stop	May	See 6.14.7.

1332 No mappings are defined for the following verbs for the specified target: `dump`, `load`, and `reset`.

1333 **6.14.1 Ordering of Results**

1334 When results are returned for multiple instances of CIM_USBRedirectionSAP, implementations shall
 1335 utilize the following algorithm to produce the natural (that is, default) ordering:

- 1336 • Results for CIM_USBRedirectionSAP are unordered; therefore, no algorithm is defined.

1337 **6.14.2 Create**

1338 This section describes how to implement the `create` verb when applied to an instance of
 1339 CIM_USBRedirectionSAP. Implementations may support the use of the `create` verb with
 1340 CIM_USBRedirectionSAP.

1341 The `create` verb is used to create a redirection session on a system.

1342 **6.14.2.1 Create Specifying Properties**

1343 **6.14.2.1.1 Command Form**

1344 `create <CIM_USBRedirectionSAP> <propertyname1>=<propertyvalue1>`
 1345 `<propertynamen>=<propertyvaluen>`

1346 **6.14.2.1.2 CIM Requirements**

1347 See CIM_USBRedirectionSAP in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
 1348 mandatory properties.

1349 **6.14.2.1.3 Behavior Requirements**

1350 **6.14.2.1.3.1 Preconditions**

1351 `$system` contains the instance of CIM_ComputerSystem that is the Resultant Target of the CLP
 1352 command. This is the container instance for the desired CIM_USBRedirectionSAP instance.

1353 **6.14.2.1.3.2 Pseudo Code**

```

1354 //1 Find the Service for the system to create the Redirection SAP
1355 #Error = &smOpAssociators (
1356     $system.getObjectPath(),
1357     "CIM_HostedService",
1358     "CIM_USBRedirectionService",
1359     NULL,
1360     NULL,
1361     NULL,
1362     $SvcInstancePaths[] )
1363 if (0 != #Error.code)
1364 {
1365     &smProcessOpError (#Error);
1366     //includes &smEnd;
1367 }
1368 $service = $SvcInstancePaths[0];
1369 //2 convert command line parms to method parms
1370 $USBSAPTemplate = smNewInstance ("CIM_USBRedirectionSAP");
1371 #CreateDevices = FALSE;
1372 $NewUSBDevice = smNewInstance ("CIM_USBDevice");
1373 $RemoteSAP = smNewInstance ("CIM_RemoteServiceAccessPoint");
1374 for #i < n
1375 {
1376     if (<proptername#i> == "NewSAPRequestedStatesSupported") {
1377         #NewRequestedStatesSupported[] = <propertyvalue#i>;
1378     }
1379     else if (<proptername#i> == "NewUSBDeviceUSBVersion") {
1380         $NewUSBDevices.USBVersion = <propertyvalue#i>;
1381         #CreateDevices = TRUE;
1382     }
1383     else if (<proptername#i> == "NewUSBDeviceClassCode") {
1384         $NewUSBDevices.ClassCode = <propertyvalue#i>;
1385         #CreateDevices = TRUE;
1386     }
1387     else if (<proptername#i> == "NewUSBDeviceSubclassCode") {
1388         $NewUSBDevices.SubclassCode = <propertyvalue#i>;
1389         #CreateDevices = TRUE;
1390     }
1391     else if (<proptername#i> == "NewUSBDeviceCommandTimeout") {
1392         $NewUSBDevices.CommandTimeout = <propertyvalue#i>;
1393         #CreateDevices = TRUE;
1394     }
1395     else if (<proptername#i> == "NewUSBDeviceProtocolCode") {
1396         $NewUSBDevices.ProtocolCode = <propertyvalue#i>;
1397         #CreateDevices = TRUE;
1398     }
1399     else if (<proptername#i> == "NewUSBDeviceMaxPacketSize") {
1400         $NewUSBDevices.MaxPacketSize = <propertyvalue#i>;

```

```

1401     #CreateDevices = TRUE;
1402 }
1403 else if (<propertname#i> == "NewUSBDeviceVendorID") {
1404     $NewUSBDevices.VendorID = <propertyvalue#i>;
1405     #CreateDevices = TRUE;
1406 }
1407 else if (<propertname#i> == "NewUSBDeviceProductID") {
1408     $NewUSBDevices.ProductID = <propertyvalue#i>;
1409     #CreateDevices = TRUE;
1410 }
1411 else if (<propertname#i> == "NewUSBDeviceDeviceReleaseNumber") {
1412     $NewUSBDevices.DeviceReleaseNumber = <propertyvalue#i>;
1413     #CreateDevices = TRUE;
1414 }
1415 else if (<propertname#i> == "NewUSBDeviceSerialNumber") {
1416     $NewUSBDevices.SerialNumber = <propertyvalue#i>;
1417     #CreateDevices = TRUE;
1418 }
1419 else if (<propertname#i> == "NewUSBDeviceManufacturer") {
1420     $NewUSBDevices.Manufacturer = <propertyvalue#i>;
1421     #CreateDevices = TRUE;
1422 }
1423 else if (<propertname#i> == "NewUSBDeviceProduct") {
1424     $NewUSBDevices.Product = <propertyvalue#i>;
1425     #CreateDevices = TRUE;
1426 }
1427 else if (<propertname#i> == "RemoteSAPAccessContext") {
1428     $RemoteSAP.AccessContext = <propertyvalue#i>;
1429 }
1430 else if (<propertname#i> == "RemoteSAPAccessInfo") {
1431     $RemoteSAP.AccessInfo = <propertyvalue#i>;
1432 }
1433 else if (<propertname#i> == "RemoteSAPInfoFormat") {
1434     $RemoteSAP.InfoFormat = <propertyvalue#i>;
1435 }
1436 else if (<propertname#i> == "RemoteSAPOtherAccessContext") {
1437     $RemoteSAP.OtherAccessContext = <propertyvalue#i>;
1438 }
1439 else if (<propertname#i> == "RemoteSAPOtherInfoFormatDescription") {
1440     $RemoteSAP.OtherInfoFormatDescription = <propertyvalue#i>;
1441 }
1442 else if (<propertname#i> == "USBDevices") {
1443 // #USBDevices[] contains references to instances of CIM_USBDevice produced
1444 // from <propertyvalue#i> using conversion mechanism presumed to exist
1445 $USBDevices[]->= <propertyvalue#i>;
1446 }
1447 else {
1448     $USBSAPTemplate.<propertname#i> = <propertyvalue#i>;
1449 }

```

```
1450 }
1451 //3 invoke the method
1452 $USBRedirectionSAP = smNewInstance ("CIM_USBRedirectionSAP");
1453 //build the parameter lists and invoke the method
1454 If (#CreateDevices) {
1455 // #NewUSBDevicesStrings contains the embedded instance produced from
1456 // $NewUSBDevices using conversion mechanism presumed to exist
1457 #NewUSBDevicesStrings[0] = $NewUSBDevices;
1458 }
1459 %InArguments[] = { newArgument("NewUSBRedirectionSAP", $USBSAPTemplate),
1460 newArgument("NewSAPRequestedStatesSupported",
1461 #NewRequestedStatesSupported[]),
1462 newArgument("USBDevices", $USBDevices[]->),
1463 newArgument("CreateDevices", $CreateDevices),
1464 newArgument("NewUSBDevices", #NewUSBDevicesStrings[]),
1465 newArgument("NewRemoteServiceAccessPoint", $RemoteSAP) };
1466 %OutArguments[] = { newArgument("SAP", $USBRedirectionSAP.GetObjectPath()) };
1467 //invoke method
1468 #returnStatus = smOpInvokeMethod ($service.GetObjectPath(),
1469 "CreateRedirectionSAP",
1470 %InArguments[],
1471 %OutArguments[]);
1472 //3 process return code to CLP Command Status
1473 if (0 != #Error.code) {
1474 //method invocation failed
1475 if ( (null != #Error.$error) && (null != #Error.$error[0]) ) {
1476 // if the method invocation contains an embedded error
1477 // use it for the Error for the overall job
1478 &smAddError($job, #Error.$error[0]);
1479 &smMakeCommandStatus($job);
1480 &smEnd;
1481 }
1482 else if (#Error.code == 17) {
1483 //trap for CIM_METHOD_NOT_FOUND
1484 //and make nice Unsupported msg.
1485 //unsupported
1486 $OperationError = smNewInstance("CIM_Error");
1487 //CIM_ERR_NOT_SUPPORTED
1488 $OperationError.CIMStatusCode = 7;
1489 //Other
1490 $OperationError.ErrorType = 1;
1491 //Low
1492 $OperationError.PerceivedSeverity = 2;
1493 $OperationError.OwningEntity = DMTF:SMCLP;
1494 $OperationError.MessageID = 0x00000001;
1495 $OperationError.Message = "Operation is not supported.";
1496 &smAddError($job, $OperationError);
1497 &smMakeCommandStatus($job);
1498 &smEnd;
```



```

1499 }
1500 else {
1501     //operation failed, but no detailed error instance, need to make one up
1502     //make an Error instance and associate with job for Operation
1503     $OperationError = smNewInstance("CIM_Error");
1504     //CIM_ERR_FAILED
1505     $OperationError.CIMStatusCode = 1;
1506     //Software Error
1507     $OperationError.ErrorType = 4;
1508     //Unknown
1509     $OperationError.PerceivedSeverity = 0;
1510     $OperationError.OwningEntity = DMTF:SMCLP;
1511     $OperationError.MessageID = 0x00000009;
1512     $OperationError.Message = "An internal software error has occurred.";
1513     &smAddError($job, $OperationError);
1514     &smMakeCommandStatus($job);
1515     &smEnd;
1516 }
1517 }//if CIM op failed
1518 else if (0 == #returnStatus) {
1519     //completed successfully, show created instance
1520     &smShowInstance($USBRedirectionSAP.getObjectPath(), NULL);
1521 &smEnd;
1522 }
1523 else if (4 == #returnStatus) {
1524     //generic failure
1525     $OperationError = smNewInstance("CIM_Error");
1526     //CIM_ERR_FAILED
1527     $OperationError.CIMStatusCode = 1;
1528     //Other
1529     $OperationError.ErrorType = 1;
1530     //Low
1531     $OperationError.PerceivedSeverity = 2;
1532     $OperationError.OwningEntity = DMTF:SMCLP;
1533     $OperationError.MessageID = 0x00000002;
1534     $OperationError.Message = "Failed. No further information is available.";
1535     &smAddError($job, $OperationError);
1536     &smMakeCommandStatus($job);
1537 }
1538 else {
1539     //invalid parameter
1540     $OperationError = smNewInstance("CIM_Error");
1541     //CIM_ERR_FAILED
1542     $OperationError.CIMStatusCode = 1;
1543     //Other
1544     $OperationError.ErrorType = 1;
1545     //Low
1546     $OperationError.PerceivedSeverity = 2;
1547     $OperationError.OwningEntity = DMTF:SMCLP;

```

```

1548     $OperationError.MessageID = 0x00000004;
1549     $OperationError.Message = "One or more parameters specified are invalid.";
1550     &smAddError($job, $OperationError);
1551     &smMakeCommandStatus($job);
1552     &smEnd;
1553 }

```

1554 6.14.3 Delete

1555 This section describes how to implement the `delete` verb when applied to an instance of
 1556 `CIM_USBRedirectionSAP`. Implementations may support the use of the `delete` verb with
 1557 `CIM_USBRedirectionSAP`.

1558 The `delete` command is used to remove an instance of `CIM_USBRedirectionSAP` that was previously
 1559 created by the `create` verb.

1560 6.14.3.1 Delete a Single Instance

1561 This command is used to delete a single instance of `CIM_USBRedirectionSAP`.

1562 6.14.3.1.1 Command Form

```

1563 Delete <CIM_USBRedirectionSAP single instance> <argumentname1>=<argumentvalue1>

```

1564 6.14.3.1.2 CIM Requirements

1565 See `CIM_USBRedirectionSAP` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
 1566 mandatory properties.

1567 6.14.3.1.3 Behavior Requirements

1568 6.14.3.1.3.1 Preconditions

1569 `$system` contains the instance of `CIM_ComputerSystem` that is the Resultant Target of the CLP
 1570 command. This is the container instance for the desired `CIM_USBRedirectionSAP` instance.

1571 6.14.3.1.3.2 Pseudo Code

```

1572 //1 Find the Service for the system to delete the Redirection SAP
1573 #Error = &smOpAssociators(
1574     $system.getObjectPath(),
1575     "CIM_HostedService",
1576     "CIM_USBRedirectionService",
1577     NULL,
1578     NULL,
1579     NULL,
1580     $SvcInstancePaths[])
1581 if (0 != #Error.code)
1582 {
1583     &smProcessOpError (#Error);
1584     //includes &smEnd;
1585 }
1586 $service = $SvcInstancePaths[0];
1587 //2 convert command line parms to method parms
1588 $instance=<CIM_RedirectionSAP single instance>;

```

```

1589 // Presumably the Delete Redirection SAP operation should always
1590 // delete all the devices created by the Create Redirection SAP
1591 // operations. So the DeleteUSBDevices parameter to the
1592 // DeleteRedirectionSAP extrinsic is always set to TRUE
1593 $DeleteDevices = TRUE;
1594 //3 invoke the method
1595 //build the parameter lists and invoke the method
1596 %InArguments[] = {newArgument("RedirectionSAP", $instance),
1597                  newArgument("DeleteUSBDevices", $DeleteDevices)};
1598 //invoke method
1599 #returnStatus = smOpInvokeMethod ($service.GetObjectPath(),
1600                                  "DeleteRedirectionSAP",
1601                                  %InArguments[],
1602                                  %OutArguments[]);
1603 //3 process return code to CLP Command Status
1604 if (0 != #Error.code) {
1605     //method invocation failed
1606     if ( (null != #Error.$error) && (null != #Error.$error[0]) ) {
1607         // if the method invocation contains an embedded error
1608         // use it for the Error for the overall job
1609         &smAddError($job, #Error.$error[0]);
1610         &smMakeCommandStatus($job);
1611         &smEnd;
1612     }
1613     else if (#Error.code == 17) {
1614         //trap for CIM_METHOD_NOT_FOUND
1615         //and make nice Unsupported msg.
1616         //unsupported
1617         $OperationError = smNewInstance("CIM_Error");
1618         //CIM_ERR_NOT_SUPPORTED
1619         $OperationError.CIMStatusCode = 7;
1620         //Other
1621         $OperationError.ErrorType = 1;
1622         //Low
1623         $OperationError.PerceivedSeverity = 2;
1624         $OperationError.OwningEntity = DMTF:SMCLP;
1625         $OperationError.MessageID = 0x00000001;
1626         $OperationError.Message = "Operation is not supported.";
1627         &smAddError($job, $OperationError);
1628         &smMakeCommandStatus($job);
1629         &smEnd;
1630     }
1631     else {
1632         //operation failed, but no detailed error instance, need to make one up
1633         //make an Error instance and associate with job for Operation
1634         $OperationError = smNewInstance("CIM_Error");
1635         //CIM_ERR_FAILED
1636         $OperationError.CIMStatusCode = 1;
1637         //Software Error

```

```
1638     $OperationError.ErrorType = 4;
1639     //Unknown
1640     $OperationError.PerceivedSeverity = 0;
1641     $OperationError.OwningEntity = DMTF:SMCLP;
1642     $OperationError.MessageID = 0x00000009;
1643     $OperationError.Message = "An internal software error has occurred.";
1644     &smAddError($job, $OperationError);
1645     &smMakeCommandStatus($job);
1646     &smEnd;
1647 }
1648 }//if CIM op failed
1649 else if (0 == #returnStatus) {
1650     //completed successfully
1651 }
1652 else if (4 == #returnStatus) {
1653     //generic failure
1654     $OperationError = smNewInstance("CIM_Error");
1655     //CIM_ERR_FAILED
1656     $OperationError.CIMStatusCode = 1;
1657     //Other
1658     $OperationError.ErrorType = 1;
1659     //Low
1660     $OperationError.PerceivedSeverity = 2;
1661     $OperationError.OwningEntity = DMTF:SMCLP;
1662     $OperationError.MessageID = 0x00000002;
1663     $OperationError.Message = "Failed. No further information is available.";
1664     &smAddError($job, $OperationError);
1665     &smMakeCommandStatus($job);
1666 }
1667 else {
1668     //invalid parameter
1669     $OperationError = smNewInstance("CIM_Error");
1670     //CIM_ERR_FAILED
1671     $OperationError.CIMStatusCode = 1;
1672     //Other
1673     $OperationError.ErrorType = 1;
1674     //Low
1675     $OperationError.PerceivedSeverity = 2;
1676     $OperationError.OwningEntity = DMTF:SMCLP;
1677     $OperationError.MessageID = 0x00000004;
1678     $OperationError.Message = "One or more parameters specified are invalid.";
1679     &smAddError($job, $OperationError);
1680     &smMakeCommandStatus($job);
1681     &smEnd;
1682 }
```

1683 6.14.4 Set

1684 The `set` verb is used to set properties on an instance of `CIM_USBRedirectionSAP`. Implementations may
1685 support the use of the `set` verb with `CIM_USBRedirectionSAP`.

1686 6.14.4.1 General Usage of Set for a Single Property

1687 This command form corresponds to the general usage of the `set` verb to modify a single property of a
1688 target instance. The setting of a single property shall be deterministic.

1689 The requirements for supporting modification of a property using this command form shall be equivalent
1690 to the requirement for supporting modification of the property using the `ModifyInstance` operation as
1691 defined in the [USB Redirection Profile](#).

1692 6.14.4.1.1 Command Form

```
1693 set <CIM_USBRedirectionSAP single object> <propertyname>=<propertyvalue>
```

1694 6.14.4.1.2 CIM Requirements

1695 See `CIM_USBRedirectionSAP` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
1696 mandatory properties.

1697 6.14.4.1.3 Behavior Requirements

1698 6.14.4.1.3.1 Pseudo Code

```
1699 $instance=<CIM_USBRedirectionSAP single object>
1700 #propertyName[] = <propertyname>
1701 #propertyValues[] = <propertyvalue>
1702 &smSetInstance ( $instance, #propertyName, #propertyValues );
1703 &smEnd;
```

1704 6.14.4.2 General Usage of Set for Multiple Properties

1705 This command form corresponds to the general usage of the `set` verb to modify multiple properties of a
1706 target instance. The setting of multiple properties may be deterministic.

1707 The requirements for supporting modification of a property using this command form shall be equivalent
1708 to the requirement for supporting modification of the property using the `ModifyInstance` operation as
1709 defined in the [USB Redirection Profile](#).

1710 6.14.4.2.1 Command Form

```
1711 set <CIM_USBRedirectionSAP multiple objects> <propertyname1>=<propertyvalue1>
1712 <propertynameN>=<propertyvalueN>
```

1713 6.14.4.2.2 CIM Requirements

1714 See `CIM_USBRedirectionSAP` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
1715 mandatory properties.

1716 6.14.4.2.3 Behavior Requirements

1717 6.14.4.2.3.1 Preconditions

1718 `$instance` represents the instance of `CIM_USBRedirectionSAP`.

1719 **6.14.4.2.3.2 Pseudo Code**

```

1720 for #i < n
1721 {
1722     #propertyName[#i] = <propertyName#i>
1723     #propertyValue[#i] = <propertyvalue#i>
1724 }
1725 &smSetInstance ( $instance, #propertyName[], #propertyValue[] );
1726 &smEnd;
```

1727 **6.14.5 Show**

1728 The `show` verb is used to display information about instances of CIM_USBRedirectionSAP.
 1729 Implementations shall support the use of the `show` verb with CIM_USBRedirectionSAP.

1730 **6.14.5.1 Show a Single Instance**

1731 This command form is used to display information about a single instance of CIM_USBRedirectionSAP.

1732 **6.14.5.1.1 Command Form**

```
1733 show <CIM_USBRedirectionSAP single instance>
```

1734 **6.14.5.1.2 CIM Requirements**

1735 See CIM_USBRedirectionSAP in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
 1736 mandatory properties.

1737 **6.14.5.1.3 Behavior Requirements**1738 **6.14.5.1.3.1 Preconditions**

1739 `$instance` represents the instance of CIM_USBRedirectionSAP.

1740 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

1741 `#propertylist[]` is an array of mandatory non-key property names.

1742 **6.14.5.1.3.2 Pseudo Code**

```

1743 if (false != #all) { #propertylist[] = NULL; }
1744 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
1745 &smEnd;
```

1746 **6.14.5.2 Show Multiple Instances – CIM_HostedAccessPoint**

1747 This command form is used to display information about multiple instances of CIM_USBRedirectionSAP.

1748 **6.14.5.2.1 Command Form**

```
1749 show <CIM_USBRedirectionSAP multiple instances>
```

1750 **6.14.5.2.2 CIM Requirements**

1751 See CIM_USBRedirectionSAP in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
 1752 mandatory properties.

1753 6.14.5.2.3 Behavior Requirements

1754 6.14.5.2.3.1 Preconditions

1755 \$containerInstance represents the instance of CIM_ComputerSystem for which the scoped instance
1756 of CIM_USBRedirectionSAP is being displayed. The CIM_USBRedirectionSAP is associated to
1757 CIM_ComputerSystem via a CIM_HostedAccessPoint association.

1758 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

1759 #propertylist[] is an array of mandatory non-key property names.

1760 6.14.5.2.3.2 Pseudo Code

```
1761 if (false != #all) { #propertylist[] = NULL; }
1762 &smShowInstances ( "CIM_USBRedirectionSAP", "CIM_HostedAccessPoint",
1763     $containerInstance.GetObjectPath(), #propertylist[] );
1764 &smEnd;
```

1765 6.14.5.3 Show Multiple Instances – CIM_SAPAvailableForElement

1766 This command form is used to display information about multiple instances of CIM_USBRedirectionSAP.

1767 6.14.5.3.1 Command Form

```
1768 show <CIM_USBRedirectionSAP multiple instances>
```

1769 6.14.5.3.2 CIM Requirements

1770 See CIM_USBRedirectionSAP in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
1771 mandatory properties.

1772 6.14.5.3.3 Behavior Requirements

1773 6.14.5.3.3.1 Preconditions

1774 \$containerInstance represents the instance of CIM_ComputerSystem for which the scoped instance
1775 of CIM_USBRedirectionSAP is being displayed. The CIM_USBRedirectionSAP is associated to
1776 CIM_ComputerSystem via a CIM_SAPAvailableForElement association.

1777 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

1778 #propertylist[] is an array of mandatory non-key property names.

1779 6.14.5.3.3.2 Pseudo Code

```
1780 if (false != #all) { #propertylist[] = NULL; }
1781 &smShowInstances ( "CIM_USBRedirectionSAP", "CIM_SAPAvailableForElement",
1782     $containerInstance.GetObjectPath(), #propertylist[] );
1783 &smEnd;
```

1784 6.14.5.4 Show Multiple Instances – CIM_ServiceAccessBySAP

1785 This command form is used to display information about multiple instances of CIM_USBRedirectionSAP.

1786 6.14.5.4.1 Command Form

```
1787 show <CIM_USBRedirectionSAP multiple instances>
```

1788 **6.14.5.4.2 CIM Requirements**

1789 See CIM_USBRedirectionSAP in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
1790 mandatory properties.

1791 **6.14.5.4.3 Behavior Requirements**1792 **6.14.5.4.3.1 Preconditions**

1793 \$containerInstance represents the instance of CIM_USBRedirectionService for which the scoped
1794 instance of CIM_USBRedirectionSAP is being displayed. The CIM_USBRedirectionSAP is associated to
1795 CIM_USBRedirectionService via a CIM_ServiceAccessBySAP association.

1796 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

1797 #propertylist[] is an array of mandatory non-key property names.

1798 **6.14.5.4.3.2 Pseudo Code**

```
1799 if (false != #all) { #propertylist[] = NULL; }
1800 &smShowInstances ( "CIM_USBRedirectionSAP", "CIM_ServiceAccessBySAP",
1801     $containerInstance.getObjectPath(), #propertylist[] );
1802 &smEnd;
```

1803 **6.14.6 Start**

1804 This section describes how to implement the `start` verb when applied to an instance of
1805 CIM_USBRedirectionSAP. Implementations may support the use of the `start` verb with
1806 CIM_USBRedirectionSAP.

1807 The `start` verb is used to enable a USB Redirection Session.

1808 **6.14.6.1 Start a Single Instance**

1809 This command form is for the `start` verb applied to a single instance of CIM_USBRedirectionSAP, in
1810 which the CIM_USBRedirectionSAP does not use CLP session for the USB Redirection session.

1811 **6.14.6.1.1 Command Form**

```
1812 start <CIM_USBRedirectionSAP single instance>
```

1813 **6.14.6.1.2 CIM Requirements**

1814 See CIM_USBRedirectionSAP in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
1815 mandatory properties.

1816 **6.14.6.1.3 Behavior Requirements**

```
1817 $instance=<CIM_USBRedirectionSAP single instance>
1818 &smStartRSC ( $instance.getObjectPath() );
1819 &smEnd;
```

1820 **6.14.7 Stop**

1821 This section describes how to implement the `stop` verb when applied to an instance of
1822 CIM_USBRedirectionSAP. Implementations may support the use of the `stop` verb with
1823 CIM_USBRedirectionSAP.

1824 The `stop` verb is used to place a USB Redirection session into a 6 (Enabled but Offline) state.

1825 **6.14.7.1 Stop a Single Instance**

1826 This command form is for the `stop` verb applied to a single instance of `CIM_USBRedirectionSAP`.

1827 **6.14.7.1.1 Command Form**

```
1828 stop <CIM_USBRedirectionSAP single instance>
```

1829 **6.14.7.1.2 CIM Requirements**

1830 See `CIM_USBRedirectionSAP` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
1831 mandatory properties.

1832 **6.14.7.1.3 Behavior Requirements**

1833 **6.14.7.1.3.1 Preconditions**

1834 `$instance` represents the targeted instance of `CIM_USBRedirectionSAP`.

1835 **6.14.7.1.3.2 Pseudo Code**

```
1836 $instance=<CIM_USBRedirectionSAP single instance>  
1837 &smRequestStateChange ( $instance.getObjectPath(), "Offline" );  
1838 &smEnd;
```

1839 **6.14.7.2 Set RequestedState to “Disabled”**

1840 This section describes how to change the state of the Redirection Session represented by
1841 `CIM_USBRedirectionSAP` to “Disabled”.

1842 **6.14.7.2.1 Command Form**

```
1843 set <CIM_USBRedirectionSAP single instance> RequestedState="Disabled"
```

1844 **6.14.7.2.2 CIM Requirements**

1845 See `CIM_USBRedirectionSAP` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list of
1846 mandatory properties.

1847 **6.14.7.2.3 Behavior Requirements**

1848 **6.14.7.2.3.1 Preconditions**

1849 `$instance` represents the targeted instance of `CIM_USBRedirectionSAP`.

1850 **6.14.7.2.3.2 Pseudo Code**

```
1851 $instance=<CIM_USBRedirectionSAP single instance>  
1852 &smStopRSC ( $instance.getObjectPath() );  
1853 &smEnd;
```

1854 **6.15 CIM_USBRedirectionService**

1855 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

1856 Table 15 lists each SM CLP verb, the required level of support for the verb in conjunction with the target
1857 class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and

1858 target. Table 15 is for informational purposes only; in case of a conflict between Table 15 and
 1859 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 1860 information in Table 15.

1861 **Table 15 – Command Verb Requirements for CIM_USBRedirectionService**

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

1862 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 1863 and `reset`.

1864 6.15.1 Ordering of Results

1865 When results are returned for multiple instances of `CIM_USBRedirectionService`, implementations shall
 1866 utilize the following algorithm to produce the natural (that is, default) ordering:

- 1867 • Results for `CIM_USBRedirectionService` are unordered; therefore, no algorithm is defined.

1868 6.15.2 Set

1869 This section describes how to implement the `set` verb when it is applied to an instance of
 1870 `CIM_USBRedirectionService`. The `set` verb is used to set properties on an instance of
 1871 `CIM_USBRedirectionService`.

1872 Implementations may support the use of the `set` verb with `CIM_USBRedirectionService`.

1873 6.15.2.1 General Usage of Set for a Single Property

1874 This command form corresponds to the general usage of the `set` verb to modify a single property of a
 1875 target instance. The setting of a single property may be deterministic.

1876 The requirements for supporting modification of a property using this command form shall be equivalent
 1877 to the requirement for supporting modification of the property using the `ModifyInstance` operation as
 1878 defined in the [USB Redirection Profile](#).

1879 6.15.2.1.1 Command Form

1880 `set <CIM_USBRedirectionService single object> <propertyname>=<propertyvalue>`

1881 6.15.2.1.2 CIM Requirements

1882 See `CIM_USBRedirectionService` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list
 1883 of mandatory properties.

1884 6.15.2.1.3 Behavior Requirements

1885 6.15.2.1.3.1 Pseudo Code

```

1886 $instance=<CIM_USBRedirectionService single object>
1887 #propertyName[] = <propertyName>
1888 #propertyValues[] = <propertyvalue>
1889 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
1890 &smEnd;

```

1891 6.15.2.2 General Usage of Set for Multiple Properties

1892 This command form corresponds to the general usage of the `set` verb to modify multiple properties of a
 1893 target instance. The setting of multiple properties may be deterministic.

1894 The requirements for supporting modification of a property using this command form shall be equivalent
 1895 to the requirement for supporting modification of the property using the `ModifyInstance` operation as
 1896 defined in the [USB Redirection Profile](#).

1897 6.15.2.2.1 Command Form

```

1898 set <CIM_USBRedirectionService multiple objects> <propertyName1>=<propertyvalue1>
1899 <propertyNameN>=<propertyvalueN>

```

1900 6.15.2.2.2 CIM Requirements

1901 See `CIM_USBRedirectionService` in the “CIM Elements” section of the [USB Redirection Profile](#) for the list
 1902 of mandatory properties.

1903 6.15.2.2.3 Behavior Requirements

1904 6.15.2.2.3.1 Preconditions

1905 `$instance` represents the instance of `CIM_USBRedirectionService`.

1906 6.15.2.2.3.2 Pseudo Code

```

1907 for #i < n
1908 {
1909   #propertyName[#i] = <propertyName#i>
1910   #propertyValues[#i] = <propertyvalue#i>
1911 }
1912 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
1913 &smEnd;

```

1914 6.15.3 Show

1915 The `show` verb is used to display information about instances of `CIM_USBRedirectionService`.
 1916 Implementations shall support the use of the `show` verb with `CIM_USBRedirectionService`.

1917 6.15.3.1 Show Command Form for a Single Instance

1918 This command form is to show a single instance of `CIM_USBRedirectionService`.

1919 6.15.3.1.1 Command Form

```

1920 show <CIM_USBRedirectionService single instance>

```

1921 6.15.3.1.2 CIM Requirements

1922 See CIM_USBRedirectionService in the “CIM Elements” section of the [USB Redirection Profile](#) for the list
1923 of mandatory properties.

1924 6.15.3.1.3 Behavior Requirements

1925 6.15.3.1.3.1 Preconditions

1926 `$instance` represents the instance of CIM_USBRedirectionService.

1927 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

1928 `#propertylist[]` is an array of mandatory non-key property names.

1929 6.15.3.1.3.2 Pseudo Code

```
1930 if (false != #all) { #propertylist[] = NULL; }  
1931 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );  
1932 &smEnd;
```

1933 6.15.4 Start

1934 6.15.4.1 Start Command Form for a Single Instance

1935 This command form is for the `start` verb applied to a single instance of CIM_USBRedirectionService.
1936 Implementations may support the use of the `start` verb with CIM_USBRedirectionService.

1937 The `start` verb is used to enable a USB Redirection Service.

1938 6.15.4.1.1 Command Form

```
1939 start <CIM_USBRedirectionService single instance>
```

1940 6.15.4.1.2 CIM Requirements

1941 See CIM_USBRedirectionService in the “CIM Elements” section of the [USB Redirection Profile](#) for the list
1942 of mandatory properties.

1943 6.15.4.1.3 Behavior Requirements

```
1944 $instance=<CIM_USBRedirectionService single instance>  
1945 &smStartRSC($instance.getObjectPath());  
1946 &smEnd;
```

1947 6.15.5 Stop

1948 This section describes how to implement the `stop` verb when applied to an instance of
1949 CIM_USBRedirectionService. Implementations may support the use of the `stop` verb with
1950 CIM_USBRedirectionService.

1951 The `stop` verb is used to place a USB Redirection Service in a 3 (Disabled) state.

1952 6.15.5.1 Stop Command Form for a Single Instance

1953 This command form is for the `stop` verb applied to a single instance of CIM_USBRedirectionService.

1954 **6.15.5.1.1 Command Form**1955 `stop <CIM_USBRedirectionService single instance>`1956 **6.15.5.1.2 CIM Requirements**

1957 See CIM_USBRedirectionService in the “CIM Elements” section of the [USB Redirection Profile](#) for the list
1958 of mandatory properties.

1959 **6.15.5.1.3 Behavior Requirements**

```
1960 $instance=<CIM_USBRedirectionService single instance>;  
1961 &smStopRSC ( $instance.GetObjectPath() );  
1962 &smEnd;
```

1963

1964
1965
1966
1967
1968

ANNEX A
(informative)

Change Log

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

1969