



1
2
3
4

Document Number: DSP0802

Date: 2009-06-04

Version: 1.0.0

5
6

SMASH Collections Profile SM CLP Command Mapping Specification

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

10

11 Copyright notice

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

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

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27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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

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

33

34

CONTENTS

35 Foreword 5

36 Introduction 6

37 1 Scope 7

38 2 Normative References..... 7

39 2.1 Approved 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_ConcreteCollection 9

45 6.2 CIM_MemberOfCollection 11

46 6.3 CIM_OwningCollectionElement 14

47 ANNEX A (informative) Change Log 17

48

49 Tables

50 Table 1 – Command Verb Requirements for CIM_ConcreteCollection 10

51 Table 2 – Command Verb Requirements for CIM_MemberOfCollection 11

52 Table 3 – Command Verb Requirements for CIM_OwningCollectionElement 14

53

55

Foreword

56 The *SMASH Collections Profile SM CLP Command Mapping Specification* (DSP0802) was prepared by
57 the Server Management Working Group.

58 **Conventions**

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

61 **Acknowledgements**

62 The authors wish to acknowledge the following participants from the DMTF Server Management Working
63 Group:

- 64 • Aaron Merkin – IBM
- 65 • Jon Hass – Dell
- 66 • Khachatur Papanyan – Dell
- 67 • Jeff Hilland – HP
- 68 • Christina Shaw – HP
- 69 • Perry Vincent – Intel
- 70 • John Leung – Intel

71

72

Introduction

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

77 The target audience for this specification is implementers of the SM CLP support for the [SMASH](#)
78 [Collections Profile](#).

79 SMASH Collections Profile SM CLP Command Mapping 80 Specification

81 1 Scope

82 This specification contains the requirements for an implementation of the SM CLP to provide access to,
83 and implement the behaviors of, the [SMASH Collections Profile](#).

84 2 Normative References

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

88 2.1 Approved References

89 DMTF DSP1006, *SMASH Collections Profile 1.0*,
90 http://www.dmtf.org/standards/published_documents/DSP1006_1.0.pdf

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

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

95 Other References

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

98 3 Terms and Definitions

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

100 3.1

101 **can**

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

103 3.2

104 **cannot**

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

106 3.3

107 **conditional**

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

- 110 **3.4**
111 **mandatory**
112 indicates requirements to be followed strictly in order to conform to the document and from which no
113 deviation is permitted
- 114 **3.5**
115 **may**
116 indicates a course of action permissible within the limits of the document
- 117 **3.6**
118 **need not**
119 indicates a course of action permissible within the limits of the document
- 120 **3.7**
121 **optional**
122 indicates a course of action permissible within the limits of the document
- 123 **3.8**
124 **shall**
125 indicates requirements to be followed strictly in order to conform to the document and from which no
126 deviation is permitted
- 127 **3.9**
128 **shall not**
129 indicates requirements to be followed strictly in order to conform to the document and from which no
130 deviation is permitted
- 131 **3.10**
132 **should**
133 indicates that among several possibilities, one is recommended as particularly suitable, without
134 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 135 **3.11**
136 **should not**
137 indicates that a certain possibility or course of action is deprecated but not prohibited

138 **4 Symbols and Abbreviated Terms**

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

- 140 **4.1**
141 **CIM**
142 Common Information Model
- 143 **4.2**
144 **CLP**
145 Command Line Protocol
- 146 **4.3**
147 **DMTF**
148 Distributed Management Task Force

149 **4.4**
150 **IETF**
151 Internet Engineering Task Force

152 **4.5**
153 **SM**
154 Server Management

155 **4.6**
156 **SMI-S**
157 Storage Management Initiative Specification

158 **4.7**
159 **SNIA**
160 Storage Networking Industry Association

161 **5 Recipes**

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

- 164 • smShowInstance()
- 165 • smShowInstances()
- 166 • smSetInstance()
- 167 • smShowAssociationInstances()
- 168 • smShowAssociationInstance()

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

170 **6 Mappings**

171 The following sections detail the mapping of CLP verbs to CIM Operations for each CIM class defined in
172 the *SMASH Collections Profile* ([DSP1006](#)). Requirements specified here related to support for a CLP
173 verb for a particular class are solely within the context of this profile.

174 **6.1 CIM_ConcreteCollection**

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

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

181

Table 1 – Command Verb Requirements for CIM_ConcreteCollection

Command Verb	Requirement	Comments
create	Not supported	None
delete	Not supported	None
dump	Not supported	None
load	Not supported	None
reset	Not supported	None
set	Not supported	None
show	Shall	See 6.1.2
start	Not supported	None
stop	Not supported	None

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

184 6.1.1 Ordering of Results

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

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

188 6.1.2 Show

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

192 6.1.2.1 Show a Single Instance of CIM_ConcreteCollection

193 6.1.2.1.1 Command Form

```
194 show <CIM_ConcreteCollection single instance>
```

195 6.1.2.1.2 CIM Requirements

196 See the “CIM Elements” section of the [SMASH Collections Profile](#).

197 6.1.2.1.3 Behavior Requirements

198 6.1.2.1.3.1 Preconditions

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

200 6.1.2.1.3.2 Pseudo Code

```
201 $instance=<CIM_ConcreteCollection single instance>
202 #propertylist[] = NULL;
203 if (false == #all) {
204     #propertylist[] = { //all mandatory non-key properties }
205 }
206 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
207 &smEnd;
```

208 **6.1.2.2 Show Multiple Instances of CIM_ConcreteCollection**

209 **6.1.2.2.1 Command Form**

210 `show <CIM_ConcreteCollection multiple instances>`

211 **6.1.2.2.2 CIM Requirements**

212 See the “CIM Elements” section of the [SMASH Collections Profile](#).

213 **6.1.2.2.3 Behavior Requirements**

214 **6.1.2.2.3.1 Preconditions**

215 \$containerInstance contains the instance of CIM_ComputerSystem for which we are displaying
216 related CIM_ConcreteCollection instances.

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

218 **6.1.2.2.3.2 Pseudo Code**

```
219 #propertylist[] = NULL;
220 if (false == #all) {
221     #propertylist[] = { //all mandatory non-key properties };
222 }
223 &smShowInstances ( "CIM_ConcreteCollection", "CIM_OwningCollectionElement",
224     $containerInstance.getObjectPath(), #propertylist[] );
225 &smEnd;
```

226 **6.2 CIM_MemberOfCollection**

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

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

233 **Table 2 – Command Verb Requirements for CIM_MemberOfCollection**

Command Verb	Requirement	Comments
create	Not supported	None
delete	Not supported	None
dump	Not supported	None
load	Not supported	None
reset	Not supported	None
set	Not supported	None
show	Shall	See 6.2.2
start	Not supported	None
stop	Not supported	None

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

236 6.2.1 Ordering of Results

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

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

240 6.2.2 Show

241 This section describes how to implement the `show` verb when applied to an instance of
242 CIM_MemberOfCollection. Implementations shall support the use of the `show` verb with
243 CIM_MemberOfCollection.

244 6.2.2.1 Show a Single Instance – Both References

245 This command form is for the `show` command applied to CIM_MemberOfCollection where both
246 references are specified. Therefore, exactly one instance is shown.

247 6.2.2.1.1 Command Form

```
248 show <CIM_MemberOfCollection single instance>
```

249 6.2.2.1.2 CIM Requirements

250 See the “CIM Elements” section of the [SMASH Collections Profile](#).

251 6.2.2.1.3 Behavior Requirements

252 6.2.2.1.3.1 Preconditions

253 `$instanceA` contains one of the instances of CIM_ConcreteCollection referenced by
254 CIM_MemberOfCollection.

255 `$instanceB` contains one of the instances of CIM_ManagedElement referenced by
256 CIM_MemberOfCollection.

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

258 6.2.2.1.3.2 Psuedo Code

```
259 #propertylist[] = NULL;  
260 if ( false == #all) {  
261     #propertylist[] = { //all mandatory non-key properties };  
262 }  
263 &smShowAssociationInstance ( "CIM_MemberOfCollection", $instanceA.getObjectPath(),  
264     $instanceB.getObjectPath(), #propertylist[] );  
265 &smEnd;
```

266 6.2.2.2 Show Multiple Instances – CIM_ConcreteCollection Reference

267 This command form is for the `show` command applied to an instance of CIM_MemberOfCollection where
268 only the reference to an instance of CIM_ConcreteCollection is specified. Zero or more instances of
269 CIM_MemberOfCollection can reference a single instance of CIM_ConcreteCollection.

270 6.2.2.2.1 Command Form

```
271 show <CIM_MemberOfCollection multiple instances>
```

272 6.2.2.2.2 CIM Requirements

273 See the “CIM Elements” section of the [SMASH Collections Profile](#).

274 6.2.2.2.3 Behavior Requirements

275 6.2.2.2.3.1 Preconditions

276 \$instance contains the instance of CIM_ConcreteCollection that is referenced by
277 CIM_MemberOfCollection

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

279 6.2.2.2.3.2 Psuedo Code

```
280 #propertylist[] = NULL;
281 if ( false == #all) {
282     #propertylist[] = { //all mandatory non-key properties};
283 }
284 &smShowAssociationInstances ( "CIM_MemberOfCollection", $instance.getObjectPath(),
285     #propertylist[] );
286 &smEnd;
```

287 6.2.2.3 Show a Single Instance – CIM_ManagedElement Reference

288 This command form is for the show command applied to CIM_MemberOfCollection where only the
289 reference to an instance of CIM_ManagedElement is specified. An instance of CIM_ManagedElement
290 can be referenced by at most one instance of CIM_MemberOfCollection.

291 6.2.2.3.1 Command Form

```
292 show <CIM_MemberOfCollection single instances>
```

293 6.2.2.3.2 CIM Requirements

294 See the “CIM Elements” section of the [SMASH Collections Profile](#).

295 6.2.2.3.3 Behavior Requirements

296 6.2.2.3.3.1 Preconditions

297 \$instance contains the instance of CIM_ManagedElement that is referenced by
298 CIM_MemberOfCollection

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

300 6.2.2.3.3.2 Psuedo Code

```
301 #propertylist[] = NULL;
302 if ( false == #all) {
303     #propertylist[] = { //all mandatory non-key properties};
304 }
305 &smShowAssociationInstances ( "CIM_MemberOfCollection", $instance.getObjectPath(),
306     #propertylist[] );
307 &smEnd;
```

308 6.3 CIM_OwningCollectionElement

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

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

315 **Table 3 – Command Verb Requirements for CIM_OwningCollectionElement**

Command Verb	Requirement	Comments
create	Not supported	None
delete	Not supported	None
dump	Not supported	None
load	Not supported	None
reset	Not supported	None
set	Not supported	None
show	Shall	See 6.3.2.
start	Not supported	None
stop	Not supported	None

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

318 6.3.1 Ordering of Results

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

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

322 6.3.2 Show

323 This section describes how to implement the `show` verb when applied to an instance of
 324 `CIM_OwningCollectionElement`. Implementations shall support the use of the `show` verb with
 325 `CIM_OwningCollectionElement`.

326 6.3.2.1 Show a Single Instance – Both References

327 This command form is for the `show` command applied to `CIM_OwningCollectionElement` where both
 328 references are specified. Therefore, a single instance will be returned.

329 6.3.2.1.1 Command Form

330 `show <CIM_OwningCollectionElement single instance>`

331 6.3.2.1.2 CIM Requirements

332 See the “CIM Elements” section of the [SMASH Collections Profile](#).

333 6.3.2.1.3 Behavior Requirements

334 6.3.2.1.3.1 Preconditions

335 \$instanceA contains one of the instances of <CIM_ConcreteCollection> that is referenced by
336 CIM_OwningCollectionElement

337 \$instanceB contains one of the instances of <CIM_ComputerSystem> that referenced by
338 CIM_OwningCollectionElement.

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

340 6.3.2.1.3.2 Pseudo Code

```
341 #propertylist[] = NULL;
342 if ( false == #all) {
343     #propertylist[] = { //all mandatory non-key properties};
344 }
345 &smShowAssociationInstance ( "CIM_OwningCollectionElement",
346     $instanceA.getObjectPath(), $instanceB.getObjectPath(), #propertylist[] );
347 &smEnd;
```

348 6.3.2.2 Show Multiple Instances – CIM_ComputerSystem Reference

349 This command form is for the `show` command applied to `CIM_OwningCollectionElement` where only the
350 reference to an instance of `CIM_ComputerSystem` is specified. An instance of `CIM_ComputerSystem` can
351 be referenced by multiple instances of `CIM_OwningCollectionElement`.

352 6.3.2.2.1 Command Form

```
353 show <CIM_OwningCollectionElement multiple instances>
```

354 6.3.2.2.2 CIM Requirements

355 See the “CIM Elements” section of the [SMASH Collections Profile](#).

356 6.3.2.2.3 Behavior Requirements

357 6.3.2.2.3.1 Preconditions

358 \$instance contains the instance of <CIM_ComputerSystem> that is referenced by
359 CIM_OwningCollectionElement

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

361 6.3.2.2.3.2 Pseudo Code

```
362 #propertylist[] = NULL;
363 if ( false == #all) {
364     #propertylist[] = { //all mandatory non-key properties};
365 }
366 &smShowAssociationInstances ( "CIM_OwningCollectionElement",
367     $instance.getObjectPath(), #propertylist[] );
368 &smEnd;
```

369 6.3.2.3 Show a Single Instance – CIM_ConcreteCollection Reference

370 This command form is for the `show` command applied to `CIM_OwningCollectionElement` where only the
371 reference to an instance of `CIM_ConcreteCollection` is specified. An instance of `CIM_ConcreteCollection`
372 is referenced by exactly one instance of `CIM_OwningCollectionElement`. Therefore, a single instance is
373 returned.

374 6.3.2.3.1 Command Form

```
375 show <CIM_OwningCollectionElement single instances>
```

376 6.3.2.3.2 CIM Requirements

377 See the “CIM Elements” section of the [SMASH Collections Profile](#).

378 6.3.2.3.3 Behavior Requirements

379 6.3.2.3.3.1 Preconditions

380 `$instance` contains the instance of `<CIM_ConcreteCollection>` that is referenced by
381 `CIM_OwningCollectionElement`

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

383 6.3.2.3.3.2 Pseudo Code

```
384 #propertylist[] = NULL;  
385 if ( false == #all) {  
386     #propertylist[] = { //all mandatory non-key properties};  
387 }  
388 &smShowAssociationInstances ( "CIM_OwningCollectionElement",  
389     $instance.getObjectPath(), #propertylist[] );  
390 &smEnd;
```

391

392
 393
 394
 395
 396

ANNEX A
 (informative)

Change Log

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

397