



1  
2  
3  
4

**Document Number: DSP0845**

**Date: 2009-06-04**

**Version: 1.0.0**

5 **Base Metrics Profile SM CLP Command Mapping**  
6 **Specification**

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

10

11 Copyright notice

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

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

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27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party  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

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

33

# CONTENTS

34 Foreword ..... 5

35 Introduction ..... 6

36 1 Scope ..... 7

37 2 Normative References..... 7

38 2.1 Approved References ..... 7

39 2.1 Other References..... 7

40 3 Terms and Definitions..... 7

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

42 5 Recipes..... 9

43 5.1 Local Recipes ..... 9

44 6 Mappings..... 14

45 6.1 CIM\_OperatingSystem..... 14

46 6.2 CIM\_HostedService ..... 17

47 6.3 CIM\_MetricInstance ..... 19

48 6.4 CIM\_ServiceAffectsElement ..... 21

49 6.5 CIM\_MetricDefForME ..... 24

50 6.6 CIM\_MetricsForME ..... 26

51 6.7 CIM\_ConcreteDependency ..... 28

52 6.8 CIM\_BaseMetricDefinition ..... 31

53 6.9 CIM\_BaseMetricValue ..... 38

54 6.10 CIM\_AggregationMetricDefinition ..... 41

55 6.11 CIM\_AggregationMetricValue ..... 47

56 6.12 CIM\_MetricServiceCapabilities ..... 50

57 6.13 CIM\_MetricService..... 52

58 ANNEX A (informative) Change Log..... 56

59

## 60 Tables

61 Table 1 – Command Verb Requirements for CIM\_OperatingSystem..... 15

62 Table 2 – Command Verb Requirements for CIM\_HostedService ..... 17

63 Table 3 – Command Verb Requirements for CIM\_MetricInstance ..... 19

64 Table 4 – Command Verb Requirements for CIM\_ServiceAffectsElement ..... 22

65 Table 5 – Command Verb Requirements for CIM\_MetricDefForME ..... 24

66 Table 6 – Command Verb Requirements for CIM\_MetricsForME ..... 26

67 Table 7 – Command Verb Requirements for CIM\_ConcreteDependency ..... 28

68 Table 8 – Command Verb Requirements for CIM\_BaseMetricDefinition ..... 32

69 Table 9 – Command Verb Requirements for CIM\_BaseMetricValue ..... 38

70 Table 10 – Command Verb Requirements for CIM\_AggregationMetricDefinition ..... 41

71 Table 11 – Command Verb Requirements for CIM\_AggregationMetricValue ..... 47

72 Table 12 – Command Verb Requirements for CIM\_MetricServiceCapabilities ..... 50

73 Table 13 – Command Verb Requirements for CIM\_MetricService..... 52

74



76

## Foreword

77 The *Base Metrics Profile SM CLP Command Mapping Specification* (DSP0845) was prepared by the  
78 Server Management Working Group.

### 79 **Conventions**

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

### 82 **Acknowledgements**

- 83 • Khachatur Papanyan – Dell

84

85

## Introduction

86 This document defines the SM CLP mapping for CIM elements described in the [Base Metrics Profile](#). The  
87 information in this specification, combined with the *SM CLP-to-CIM Common Mapping Specification 1.0*  
88 ([DSP0216](#)), is intended to be sufficient to implement SM CLP commands relevant to the classes,  
89 properties, and methods described in the [Base Metrics Profile](#) using CIM operations.

90 The target audience for this specification is implementers of the SM CLP support for the [Base Metrics](#)  
91 [Profile](#).

# 92 Base Metrics Profile SM CLP Command Mapping 93 Specification

## 94 1 Scope

95 This specification contains the requirements for an implementation of the SM CLP to provide access to,  
96 and implement the behaviors of, the [Base Metrics Profile](#).

## 97 2 Normative References

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

### 101 2.1 Approved References

102 DMTF DSP1053, *Base Metrics Profile 1.0*,  
103 [http://www.dmtf.org/standards/published\\_documents/DSP1053\\_1.0.pdf](http://www.dmtf.org/standards/published_documents/DSP1053_1.0.pdf)

104 DMTF DSP0216, *SM CLP-to-CIM Common Mapping Specification 1.0*,  
105 [http://www.dmtf.org/standards/published\\_documents/DSP0216\\_1.0.pdf](http://www.dmtf.org/standards/published_documents/DSP0216_1.0.pdf)

106 SNIA, *Storage Management Initiative Specification (SMI-S) 1.1.0*,  
107 [http://www.snia.org/tech\\_activities/standards/curr\\_standards/smi](http://www.snia.org/tech_activities/standards/curr_standards/smi)

### 108 2.1 Other References

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

## 111 3 Terms and Definitions

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

### 113 3.1

#### 114 **can**

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

### 116 3.2

#### 117 **cannot**

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

### 119 3.3

#### 120 **conditional**

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

- 123 **3.4**  
124 **mandatory**  
125 indicates requirements to be followed strictly in order to conform to the document and from which no  
126 deviation is permitted
- 127 **3.5**  
128 **may**  
129 indicates a course of action permissible within the limits of the document
- 130 **3.6**  
131 **need not**  
132 indicates a course of action permissible within the limits of the document
- 133 **3.7**  
134 **optional**  
135 indicates a course of action permissible within the limits of the document
- 136 **3.8**  
137 **shall**  
138 indicates requirements to be followed strictly in order to conform to the document and from which no  
139 deviation is permitted
- 140 **3.9**  
141 **shall not**  
142 indicates requirements to be followed strictly in order to conform to the document and from which no  
143 deviation is permitted
- 144 **3.10**  
145 **should**  
146 indicates that among several possibilities, one is recommended as particularly suitable, without  
147 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 148 **3.11**  
149 **should not**  
150 indicates that a certain possibility or course of action is deprecated but not prohibited

## 151 **4 Symbols and Abbreviated Terms**

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

- 153 **4.1**  
154 **CIM**  
155 Common Information Model
- 156 **4.2**  
157 **CLP**  
158 Command Line Protocol
- 159 **4.3**  
160 **DMTF**  
161 Distributed Management Task Force



- 162 **4.4**
- 163 **SM**
- 164 Server Management
- 165 **4.5**
- 166 **SMI-S**
- 167 Storage Management Initiative Specification
- 168 **4.6**
- 169 **SNIA**
- 170 Storage Networking Industry Association
- 171 **4.7**
- 172 **UFsT**
- 173 User Friendly selectionTag

## 174 **5 Recipes**

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

- 177 • smShowInstance()
- 178 • smShowInstances()
- 179 • smSetInstance()
- 180 • smShowAssociationInstance()
- 181 • smShowAssociationInstances()
- 182 • smRequestStateChange()
- 183 • smStartRSC()
- 184 • smStopRSC()

### 185 **5.1 Local Recipes**

#### 186 **5.1.1 IControlMetrics**

187 This function implements an invocation of the ControlMetrics() method on CIM\_MetricService. The  
 188 function handles the implementation through the production of Command Status.

```

189 sub void lControlMetrics($metricDef->, string #requestedState) {
190 //find associated CIM_MetricService instance and store the pointer in $service
191 #Error = &smOpAssociators ($metricDef->, "CIM_ServiceAffectsElement",
192 "Dell_OEMPowerUtilizationManagementService", "AffectedElement",
193 "AffectingElement", NULL, $outInstancePaths->[]);
194 if (0 != #Error.code)
195 {
196     &smProcessOpError (#Error);
197     //includes &smEnd;
198 }
199 if ( 0 < $outInstancePaths.length ) {
200     $service = $outInstancePaths->[0];
201 }
  
```

```
202 else {
203     // generic failure
204     $OperationError = smNewInstance("CIM_Error");
205     //CIM_ERR_FAILED
206     $OperationError.CIMStatusCode = 1;
207     //Other
208     $OperationError.ErrorType = 1;
209     //Low
210     $OperationError.PerceivedSeverity = 2;
211     $OperationError.OwningEntity = DMTF:SMCLP;
212     $OperationError.MessageID = 0x00000002;
213     $OperationError.Message = "Failed. No further information is available.";
214     &smAddError($job, $OperationError);
215     &smMakeCommandStatus($job);
216     &smEnd;
217 }
218 #intRequestedState = <integer value of valuemap string #requestedState>
219 %InArguments[] = {newArgument("Subject", NULL),
220     newArgument("Definition", $metricDef->),
221     newArgument("MetricCollectionEnabled", #intRequestedState) };
222 %OutArguments[] = {};
223 #Error = smOpInvokeMethod ($service->,
224     "ControlMetrics",
225     %InArguments[],
226     %OutArguments[],
227     #returnStatus);
228 if (0 != #Error.code) {
229     //method invocation failed
230     if ( (null != #Error.$error) && (null != #Error.$error[0]) ) {
231         //if the method invocation contains an embedded error
232         //use it for the Error for the overall job
233         &smAddError($job, #Error.$error[0]);
234         &smMakeCommandStatus($job);
235         &smEnd;
236     }
237     else if (#Error.code == 17) {
238         //trap for CIM_METHOD_NOT_FOUND
239         //and make nice Unsupported msg.
240         //unsupported
241         $OperationError = smNewInstance("CIM_Error");
242         //CIM_ERR_NOT_SUPPORTED
243         $OperationError.CIMStatusCode = 7;
244         //Other
245         $OperationError.ErrorType = 1;
246         //Low
247         $OperationError.PerceivedSeverity = 2;
248         $OperationError.OwningEntity = DMTF:SMCLP;
249         $OperationError.MessageID = 0x00000001;
250         $OperationError.Message = "Operation is not supported.";
```

```

251     &smAddError($job, $OperationError);
252     &smMakeCommandStatus($job);
253     &smEnd;
254 }
255 else {
256     //operation failed, but no detailed error instance, need to make one up
257     //make an Error instance and associate with job for Operation
258     $OperationError = smNewInstance("CIM_Error");
259     //CIM_ERR_FAILED
260     $OperationError.CIMStatusCode = 1;
261     //Software Error
262     $OperationError.ErrorType = 4;
263     //Unknown
264     $OperationError.PerceivedSeverity = 0;
265     $OperationError.OwningEntity = DMTF:SMCLP;
266     $OperationError.MessageID = 0x00000009;
267     $OperationError.Message = "An internal software error has occurred.";
268     &smAddError($job, $OperationError);
269     &smMakeCommandStatus($job);
270     &smEnd;
271 }
272 else {
273     //operation failed, but no detailed error instance, need to make one up
274     //make an Error instance and associate with job for Operation
275     $OperationError = smNewInstance("CIM_Error");
276     //CIM_ERR_FAILED
277     $OperationError.CIMStatusCode = 1;
278     //Software Error
279     $OperationError.ErrorType = 4;
280     //Unknown
281     $OperationError.PerceivedSeverity = 0;
282     $OperationError.OwningEntity = DMTF:SMCLP;
283     $OperationError.MessageID = 0x00000009;
284     $OperationError.Message = "An internal software error has occurred.";
285     &smAddError($job, $OperationError);
286     &smMakeCommandStatus($job);
287     &smEnd;
288 }
289 }//if CIM op failed
290 else if (0 == #returnStatus) {
291     //completed successfully
292     &smCommandCompleted($job);
293     &smEnd;
294 }
295 else if (0x4096 == #returnStatus) {
296     //job spawned, need to watch for it to finish
297     //while the jobstate is "Running"
298     while (4 == $instanceConcreteJob.JobState){<busy wait>}
299     if (2 != $job.OperationalStatus) {

```

```
300     %InArguments[] = { }
301     %OutArguments[] = {newArgument("Job", $instanceConcreteJob.getObjectPath())}
302     #Error = smOpInvokeMethod($job,
303         "GetError"
304         %InArguments,
305         %OutArguments,
306         #returncode);
307     //Method invocation failed, internal processing error
308     if ( (0 != #Error.code) || (0 != #returncode) ) {
309     //make an Error instance and associate with job for Operation
310         $OperationError = smNewInstance("CIM_Error");
311         //CIM_ERR_FAILED
312         $OperationError.CIMStatusCode = 1;
313         //Software Error
314         $OperationError.ErrorType = 4;
315         //Unknown
316         $OperationError.PerceivedSeverity = 0;
317         $OperationError.OwningEntity = DMTF:SMCLP;
318         $OperationError.MessageID = 0x00000009;
319         $OperationError.Message = "An internal software error has occurred.";
320         &smAddError($job, $OperationError);
321         &smMakeCommandStatus($job);
322         &smEnd;
323     }
324     else {
325         //make command status
326         $joberror = %OutArguments["Error"];
327         &smCommandExecutionFailed($job, {$joberror});
328     } //end if have CIM_Error from GetError()
329 } //embedded job not OK
330 else {
331     //the job ran to completion (we assume)
332     &smCommandComplete($job);
333     &smEnd;
334 }
335 } //if job spawned
336 else if (1 == #returnStatus) {
337     //unsupported
338     $OperationError = smNewInstance("CIM_Error");
339     //CIM_ERR_NOT_SUPPORTED
340     $OperationError.CIMStatusCode = 7;
341     //Other
342     $OperationError.ErrorType = 1;
343     //Low
344     $OperationError.PerceivedSeverity = 2;
345     $OperationError.OwningEntity = DMTF:SMCLP;
346     $OperationError.MessageID = 0x00000001;
347     $OperationError.Message = "Operation is not supported.";
348     &smAddError($job, $OperationError);
```

```
349     &smMakeCommandStatus($job);
350     &smEnd;
351 }
352 else if (5 == #returnStatus) {
353     //unsupported
354     $OperationError = smNewInstance("CIM_Error");
355     //CIM_ERR_INVALID_PARAMETER
356     $OperationError.CIMStatusCode = 4;
357     //Other
358     $OperationError.ErrorType = 1;
359     //Low
360     $OperationError.PerceivedSeverity = 2;
361     $OperationError.OwningEntity = DMTF:SMCLP;
362     $OperationError.MessageID = 0x00000004;
363     $OperationError.Message = "One or more parameters specified are invalid.";
364     &smAddError($job, $OperationError);
365     &smMakeCommandStatus($job);
366     &smEnd;
367 }
368 else if (6 == #returnStatus || 4099 == #returnStatus) {
369     //busy
370     $OperationError = smNewInstance("CIM_Error");
371     //CIM_ERR_FAILED
372     $OperationError.CIMStatusCode = 1;
373     //Other
374     $OperationError.ErrorType = 1;
375     //Low
376     $OperationError.PerceivedSeverity = 2;
377     $OperationError.OwningEntity = DMTF:SMCLP;
378     $OperationError.MessageID = 0x0000000A;
379     $OperationError.Message = "The target is busy and its state cannot be
380         changed.";
381     &smAddError($job, $OperationError);
382     &smMakeCommandStatus($job);
383     &smEnd;
384 }
385 else if (4097 == $returnStatus) {
386     //invalid state transition
387     $OperationError = smNewInstance("CIM_Error");
388     //CIM_ERR_FAILED
389     $OperationError.CIMStatusCode = 1;
390     //Other
391     $OperationError.ErrorType = 1;
392     //Low
393     $OperationError.PerceivedSeverity = 2;
394     $OperationError.OwningEntity = DMTF:SMCLP;
395     $OperationError.MessageID = 0x0000000B;
396     $OperationError.Message = "The target cannot transition to the requested state
397         from its current state.";
```

```

398     &smAddError($job, $OperationError);
399     &smMakeCommandStatus($job);
400 }
401 else if ( 2 == #returnStatus || 4 == #returnStatus || 3 == $returnStatus ) {
402     //generic failure
403     $OperationError = smNewInstance("CIM_Error");
404     //CIM_ERR_FAILED
405     $OperationError.CIMStatusCode = 1;
406     //Other
407     $OperationError.ErrorType = 1;
408     //Low
409     $OperationError.PerceivedSeverity = 2;
410     $OperationError.OwningEntity = DMTF:SMCLP;
411     $OperationError.MessageID = 0x00000002;
412     $OperationError.Message = "Failed. No further information is available.";
413     &smAddError($job, $OperationError);
414     &smMakeCommandStatus($job);
415 }
416 else {
417     //unspecified return code, generic failure
418     $OperationError = smNewInstance("CIM_Error");
419     //CIM_ERR_FAILED
420     $OperationError.CIMStatusCode = 1;
421     //Other
422     $OperationError.ErrorType = 1;
423     //Low
424     $OperationError.PerceivedSeverity = 2;
425     $OperationError.OwningEntity = DMTF:SMCLP;
426     $OperationError.MessageID = 0x00000002;
427     $OperationError.Message = "Failed. No further information is available.";
428     &smAddError($job, $OperationError);
429     &smMakeCommandStatus($job);
430     &smEnd;
431 }
432 } //end smRequestStateChange()

```

## 433 6 Mappings

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

### 437 6.1 CIM\_OperatingSystem

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

439 Table 1 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of  
 440 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the  
 441 verb and target. Table 1 is for informational purposes only; in case of a conflict between Table 1 and

442 requirements detailed in the following sections, the text detailed in the following sections supersedes the  
 443 information in Table 1.

444 **Table 1 – Command Verb Requirements for CIM\_OperatingSystem**

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	

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

#### 447 **6.1.1 Ordering of Results**

448 When results are returned for multiple instances of *CIM\_ElementCapabilities*, implementations shall  
 449 utilize the following algorithm to produce the natural (that is, default) ordering:

- 450 • Results for *CIM\_ElementCapabilities* are unordered; therefore, no algorithm is defined.

#### 451 **6.1.2 Show**

452 This section describes how to implement the *show* verb when applied to an instance of  
 453 *CIM\_ElementCapabilities*. Implementations shall support the use of the *show* verb with  
 454 *CIM\_ElementCapabilities*.

##### 455 **6.1.2.1 Show Command Form for a Single Instance – CIM\_MetricService Reference**

456 This command form is used when the *show* verb applies to a single instance. This command form  
 457 corresponds to a *show* command issued against instances of *CIM\_ElementCapabilities* where only one  
 458 reference is specified and the reference is to the instance of *CIM\_MetricService*.

##### 459 **6.1.2.1.1 Command Form**

460 `show <CIM_ElementCapabilities single instance>`

##### 461 **6.1.2.1.2 CIM Requirements**

462 See *CIM\_ElementCapabilities* in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
 463 mandatory properties.

### 464 6.1.2.1.3 Behavior Requirements

#### 465 6.1.2.1.3.1 Preconditions

466 \$instance represents the instance of a CIM\_MetricService, which is referenced by  
467 CIM\_ElementCapabilities.

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

#### 469 6.1.2.1.3.2 Pseudo Code

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

### 472 6.1.2.2 Show Command Form for Multiple Instances – CIM\_MetricServiceCapabilities Reference

473 This command form is used when the show verb applies to multiple instances. This command form  
474 corresponds to a show command issued against instances of CIM\_ElementCapabilities where only one  
475 reference is specified and the reference is to the instance of CIM\_MetricServiceCapabilities.

#### 476 6.1.2.2.1 Command Form

```
477 show <CIM_ElementCapabilities multiple instances>
```

#### 478 6.1.2.2.2 CIM Requirements

479 See CIM\_ElementCapabilities in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
480 mandatory properties.

### 481 6.1.2.2.3 Behavior Requirements

#### 482 6.1.2.2.3.1 Preconditions

483 \$instance represents the instance of a CIM\_MetricServiceCapabilities, which is referenced by  
484 CIM\_ElementCapabilities.

#### 485 6.1.2.2.3.2 Pseudo Code

```
486 &smShowAssociationInstances ( "CIM_ElementCapabilities", $instance.getObjectPath(),  
487     NULL );  
488 &smEnd;
```

### 489 6.1.2.3 Show a Single Instance Target – Both References

490 This command form is used when the show verb applies to a single instance. This command form  
491 corresponds to a show command issued against instances of CIM\_ElementCapabilities where both  
492 references are specified and therefore the desired instance is unambiguously identified.

#### 493 6.1.2.3.1 Command Form

```
494 show <CIM_ElementCapabilities single instance>
```

#### 495 6.1.2.3.2 CIM Requirements

496 See CIM\_ElementCapabilities in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
497 mandatory properties.



### 498 6.1.2.3.3 Behavior Requirements

#### 499 6.1.2.3.3.1 Preconditions

500 \$instanceA represents the instance of a CIM\_MetricService which is referenced by  
501 CIM\_ElementCapabilities.

502 \$instanceB represents the instance of a CIM\_MetricServiceCapabilities or  
503 CIM\_EnabledLogicalElementCapabilities which is referenced by CIM\_ElementCapabilities.

#### 504 6.1.2.3.3.2 Pseudo Code

```
505 &smShowAssociationInstance("CIM_ElementCapabilities", $instanceA.getObjectPath(),
506     $instanceB.getObjectPath() );
507 &smEnd;
```

## 508 6.2 CIM\_HostedService

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

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

515 **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	
set	Not supported	
show	Shall	See 6.2.2.
start	Not supported	
stop	Not supported	

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

### 518 6.2.1 Ordering of Results

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

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

## 522 6.2.2 Show

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

### 525 6.2.2.1 Show Command Form for Multiple Instances – `CIM_ComputerSystem` Reference

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

#### 529 6.2.2.1.1 Command Form

```
530 show <CIM_HostedService multiple instances>
```

#### 531 6.2.2.1.2 CIM Requirements

532 See `CIM_HostedService` in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
533 mandatory properties.

#### 534 6.2.2.1.3 Behavior Requirements

##### 535 6.2.2.1.3.1 Preconditions

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

##### 537 6.2.2.1.3.2 Pseudo Code

```
538 &smShowAssociationInstances ( "CIM_HostedService", $instance.getObjectPath() );  
539 &smEnd;
```

### 540 6.2.2.2 Show Command Form for a Single Instance – `CIM_MetricService` Reference

541 This command form is used when the `show` verb applies to a single instance. The command form  
542 corresponds to the `show` verb issued against instances of `CIM_HostedService` where only one reference  
543 is specified and the reference is to an instance of `CIM_MetricService`.

#### 544 6.2.2.2.1 Command Form

```
545 show <CIM_HostedService single instance>
```

#### 546 6.2.2.2.2 CIM Requirements

547 See `CIM_HostedService` in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
548 mandatory properties.

#### 549 6.2.2.2.3 Behavior Requirements

##### 550 6.2.2.2.3.1 Preconditions

551 `$instance` represents the instance of `CIM_MetricService`, which is referenced by `CIM_HostedService`.

##### 552 6.2.2.2.3.2 Pseudo Code

```
553 &smShowAssociationInstances ( "CIM_HostedService", $instance.getObjectPath() );  
554 &smEnd;
```

555 **6.2.2.3 Show Command Form for a Single Instance – Both References**

556 This command form is used when the `show` verb applies to a single instance. This command form  
 557 corresponds to a `show` command issued against `CIM_HostedService` where both references are  
 558 specified and therefore the desired instance is unambiguously identified.

559 **6.2.2.3.1 Command Form**

```
560 show <CIM_HostedService single instance>
```

561 **6.2.2.3.2 CIM Requirements**

562 See `CIM_HostedService` in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
 563 mandatory properties.

564 **6.2.2.3.3 Behavior Requirements**

565 **6.2.2.3.3.1 Preconditions**

566 `$instanceA` represents the referenced instance of `CIM_ComputerSystem` through the `CIM_HostedService`  
 567 association.

568 `$instanceB` represents the other instance of `CIM_MetricService` which is referenced by  
 569 `CIM_HostedService`.

570 **6.2.2.3.3.2 Pseudo Code**

```
571 &smShowAssociationInstance ("CIM_HostedService", $instanceA.getObjectPath(),  
572     $instanceB.getObjectPath() );  
573 &smEnd;
```

574 **6.3 CIM\_MetricInstance**

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

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

581 **Table 3 – Command Verb Requirements for CIM\_MetricInstance**

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	

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

### 584 6.3.1 Ordering of Results

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

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

### 588 6.3.2 Show

589 This section describes how to implement the `show` verb when applied to an instance of  
590 CIM\_MetricInstance. Implementations shall support the use of the `show` verb with CIM\_MetricInstance.

#### 591 6.3.2.1 Show Command Form for Multiple Instances – CIM\_BaseMetricDefinition or 592 CIM\_AggregationMetricDefinition Reference

593 This command form is used when the `show` verb applies to multiple instances. This command form  
594 corresponds to a `show` command issued against instances of CIM\_MetricInstance where only one  
595 reference is specified and the reference is to an instance of CIM\_BaseMetricDefinition or  
596 CIM\_AggregationMetricDefinition.

##### 597 6.3.2.1.1 Command Form

```
598 show <CIM_MetricInstance multiple instances>
```

##### 599 6.3.2.1.2 CIM Requirements

600 See CIM\_MetricInstance in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
601 mandatory properties.

##### 602 6.3.2.1.3 Behavior Requirements

###### 603 6.3.2.1.3.1 Preconditions

604 `$instance` represents the instance of CIM\_BaseMetricDefinition or CIM\_AggregationMetricDefinition,  
605 which is referenced by CIM\_MetricInstance.

###### 606 6.3.2.1.3.2 Pseudo Code

```
607 &smShowAssociationInstances ( "CIM_MetricInstance", $instance.getObjectPath() );  
608 &smEnd;
```

#### 609 6.3.2.2 Show Command Form for a Single Instance – CIM\_BaseMetricValue or 610 CIM\_AggregationMetricValue Reference

611 This command form is when the `show` verb applies to a single instance. The command form corresponds  
612 to the `show` verb issued against instances of CIM\_MetricInstance where only one reference is specified  
613 and the reference is to an instance of CIM\_BaseMetricValue or CIM\_AggregationMetricValue.

##### 614 6.3.2.2.1 Command Form

```
615 show <CIM_MetricInstance single instance>
```

##### 616 6.3.2.2.2 CIM Requirements

617 See CIM\_MetricInstance in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
618 mandatory properties.

### 619 **6.3.2.2.3 Behavior Requirements**

#### 620 **6.3.2.2.3.1 Preconditions**

621 \$instance represents the instance of CIM\_BaseMetricValue or CIM\_AggregationMetricValue, which is  
622 referenced by CIM\_MetricInstance.

#### 623 **6.3.2.2.3.2 Pseudo Code**

```
624 &smShowAssociationInstances ("CIM_MetricInstance", $instance.GetObjectPath() );
625 &smEnd;
```

### 626 **6.3.2.3 Show Command Form for a Single Instance – Both References**

627 This command form is used when the `show` verb applies to a single instance. This command form  
628 corresponds to a `show` command issued against CIM\_MetricInstance where both references are  
629 specified and therefore the desired instance is unambiguously identified.

#### 630 **6.3.2.3.1 Command Form**

```
631 show <CIM_MetricInstance single instance>
```

#### 632 **6.3.2.3.2 CIM Requirements**

633 See CIM\_MetricInstance in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
634 mandatory properties.

### 635 **6.3.2.3.3 Behavior Requirements**

#### 636 **6.3.2.3.3.1 Preconditions**

637 \$instanceA represents the referenced instance of CIM\_BaseMetricDefinition or  
638 CIM\_AggregationMetricDefinition through the CIM\_MetricInstance association.

639 \$instanceB represents the other instance of CIM\_BaseMetricValue or CIM\_AggregationMetricValue  
640 which is referenced by CIM\_MetricInstance.

#### 641 **6.3.2.3.3.2 Pseudo Code**

```
642 &smShowAssociationInstance ("CIM_MetricInstance", $instanceA.GetObjectPath(),
643     $instanceB.GetObjectPath() );
644 &smEnd;
```

## 645 **6.4 CIM\_ServiceAffectsElement**

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

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

652

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	

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

#### 655 6.4.1 Ordering of Results

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

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

#### 659 6.4.2 Show

660 This section describes how to implement the *show* verb when applied to an instance of  
661 CIM\_ServiceAffectsElement. Implementations shall support the use of the *show* verb with  
662 CIM\_ServiceAffectsElement.

##### 663 6.4.2.1 Show Command Form for a Single Instance – CIM\_BaseMetricDefinition or 664 CIM\_AggregationMetricDefinition Reference

665 This command form is used when the *show* verb applies to a single instance. This command form  
666 corresponds to a *show* command issued against an instance of CIM\_ServiceAffectsElement where only  
667 one reference is specified and the reference is to an instance of CIM\_BaseMetricDefinition or  
668 CIM\_AggregationMetricDefinition.

##### 669 6.4.2.1.1 Command Form

```
670 show <CIM_ServiceAffectsElement single instance>
```

##### 671 6.4.2.1.2 CIM Requirements

672 See CIM\_ServiceAffectsElement in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
673 mandatory properties.

##### 674 6.4.2.1.3 Behavior Requirements

##### 675 6.4.2.1.3.1 Preconditions

676 \$instance represents the instance of CIM\_BaseMetricDefinition or CIM\_AggregationMetricDefinition,  
677 which is referenced by CIM\_ServiceAffectsElement.

#### 678 **6.4.2.1.3.2 Pseudo Code**

```
679 &smShowAssociationInstances ( "CIM_ServiceAffectsElement",
680     $instance.getObjectPath() );
681 &smEnd;
```

#### 682 **6.4.2.2 Show Command Form for Multiple Instances – CIM\_MetricService Reference**

683 This command form is used when the `show` verb applies to multiple instances. This command form  
684 corresponds to a `show` command issued against instances of `CIM_ServiceAffectsElement` where only  
685 one reference is specified and the reference is to an instance of `CIM_MetricService`.

##### 686 **6.4.2.2.1 Command Form**

```
687 show <CIM_ServiceAffectsElement multiple instances>
```

##### 688 **6.4.2.2.2 CIM Requirements**

689 See `CIM_ServiceAffectsElement` in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
690 mandatory properties.

##### 691 **6.4.2.2.3 Behavior Requirements**

###### 692 **6.4.2.2.3.1 Preconditions**

693 `$instance` represents the instance of `CIM_MetricService`, which is referenced by  
694 `CIM_ServiceAffectsElement`.

###### 695 **6.4.2.2.3.2 Pseudo Code**

```
696 &smShowAssociationInstances ( "CIM_ServiceAffectsElement",
697     $instance.getObjectPath() );
698 &smEnd;
```

#### 699 **6.4.2.3 Show Command Form for a Single Instance – Both References**

700 This command form is used when the `show` verb applies to a single instance. This command form  
701 corresponds to the `show` verb issued against instances of `CIM_ServiceAffectsElement` where both  
702 references are specified and therefore the desired instance is unambiguously identified.

##### 703 **6.4.2.3.1 Command Form**

```
704 show <CIM_ServiceAffectsElement single instance>
```

##### 705 **6.4.2.3.2 CIM Requirements**

706 See `CIM_ServiceAffectsElement` in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
707 mandatory properties.

##### 708 **6.4.2.3.3 Behavior Requirements**

###### 709 **6.4.2.3.3.1 Preconditions**

710 `$instanceA` represents the instance of `CIM_MetricService` which is referenced by  
711 `CIM_ServiceAffectsElement`.

712 `$instanceB` represents the instance of `CIM_BaseMetricDefinition` or `CIM_AggregationMetricDefinition`  
713 which is referenced by `CIM_ServiceAffectsElement`.

714 **6.4.2.3.3.2 Pseudo Code**

```

715 &smShowAssociationInstance ( "CIM_ServiceAffectsElement", $instanceA.getObjectPath(),
716     $instanceB.getObjectPath() );
717 &smEnd;

```

718 **6.5 CIM\_MetricDefForME**

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

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

725 **Table 5 – Command Verb Requirements for CIM\_MetricDefForME**

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 Section 6.5.2.
start	Not supported	
stop	Not supported	

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

728 **6.5.1 Ordering of Results**

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

- 731 • Results for `CIM_MetricDefForME` are unordered; therefore, no algorithm is defined.

732 **6.5.2 Show**

733 This section describes how to implement the `show` verb when applied to an instance of  
 734 `CIM_MetricDefForME`. Implementations shall support the use of the `show` verb with  
 735 `CIM_MetricDefForME`.

736 **6.5.2.1 Show Command Form for Multiple Instances – CIM\_BaseMetricDefinition or  
 737 CIM\_AggregationMetricDefinition Reference**

738 This command form is used when the `show` verb applies to multiple instances. This command form  
 739 corresponds to a `show` command issued against instances of `CIM_MetricDefForME` where only one  
 740 reference is specified and the reference is to an instance of `CIM_BaseMetricDefinition` or  
 741 `CIM_AggregationMetricDefinition`.



### 742 6.5.2.1.1 Command Form

```
743 show <CIM_MetricDefForME multiple instances>
```

### 744 6.5.2.1.2 CIM Requirements

745 See CIM\_MetricDefForME in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
746 mandatory properties.

### 747 6.5.2.1.3 Behavior Requirements

#### 748 6.5.2.1.3.1 Preconditions

749 \$instance represents the instance of CIM\_BaseMetricDefinition or CIM\_AggregationMetricDefinition,  
750 which is referenced by CIM\_MetricDefForME.

#### 751 6.5.2.1.3.2 Pseudo Code

```
752 &smShowAssociationInstances ( "CIM_MetricDefForME", $instance.getObjectPath() );  
753 &smEnd;
```

### 754 6.5.2.2 Show Command Form for Multiple Instances – CIM\_ManagedElement Reference

755 This command form is used when the `show` verb applies to multiple instances. This command form  
756 corresponds to a `show` command issued against instances of CIM\_MetricDefForME where only one  
757 reference is specified and the reference is to an instance of CIM\_ManagedElement.

#### 758 6.5.2.2.1 Command Form

```
759 show <CIM_MetricDefForME multiple instances>
```

#### 760 6.5.2.2.2 CIM Requirements

761 See CIM\_MetricDefForME in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
762 mandatory properties.

#### 763 6.5.2.2.3 Behavior Requirements

##### 764 6.5.2.2.3.1 Preconditions

765 \$instance represents the instance of CIM\_ManagedElement, which is referenced by  
766 CIM\_MetricDefForME.

##### 767 6.5.2.2.3.2 Pseudo Code

```
768 &smShowAssociationInstances ( "CIM_MetricDefForME", $instance.getObjectPath() );  
769 &smEnd;
```

### 770 6.5.2.3 Show Command Form for a Single Instance – Both References

771 This command form is used when the `show` verb applies to a single instance. This command form  
772 corresponds to the `show` verb issued against instances of CIM\_MetricDefForME where both references  
773 are specified and therefore the desired instance is unambiguously identified.

#### 774 6.5.2.3.1 Command Form

```
775 show <CIM_MetricDefForME single instance>
```

776 **6.5.2.3.2 CIM Requirements**

777 See CIM\_MetricDefForME in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
778 mandatory properties.

779 **6.5.2.3.3 Behavior Requirements**780 **6.5.2.3.3.1 Preconditions**

781 \$instanceA represents the instance of CIM\_BaseMetricDefinition or CIM\_AggregationMetricDefinition,  
782 which is referenced by CIM\_MetricDefForME.

783 \$instanceB represents the instance CIM\_ManagedElement which is referenced by CIM\_MetricDefForME.

784 **6.5.2.3.3.2 Pseudo Code**

```
785 &smShowAssociationInstance ( "CIM_MetricDefForME", $instanceA.getObjectPath(),
786     $instanceB.getObjectPath() );
787 &smEnd;
```

788 **6.6 CIM\_MetricsForME**

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

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

795 **Table 6 – Command Verb Requirements for CIM\_MetricsForME**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.6.2.
start	Not supported	
stop	Not supported	

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

798 **6.6.1 Ordering of Results**

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

- 801
- Results for CIM\_MetricForME are unordered; therefore, no algorithm is defined.

## 802 **6.6.2 Show**

803 This section describes how to implement the `show` verb when applied to an instance of  
804 `CIM_MetricForME`. Implementations shall support the use of the `show` verb with `CIM_MetricForME`.

### 805 **6.6.2.1 Show Command Form for Multiple Instances – CIM\_ManagedElement Reference**

806 This command form is used when the `show` verb applies to multiple instances. The command form  
807 corresponds to the `show` verb issued against instances of `CIM_MetricForME` where only one reference is  
808 specified and the reference is to an instance of `CIM_ManagedElement`.

#### 809 **6.6.2.1.1 Command Form**

```
810 show <CIM_MetricForME multiple instances>
```

#### 811 **6.6.2.1.2 CIM Requirements**

812 See `CIM_MetricForME` in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of mandatory  
813 properties.

#### 814 **6.6.2.1.3 Behavior Requirements**

##### 815 **6.6.2.1.3.1 Preconditions**

816 `$instance` represents the instance of `CIM_ManagedElement`, which is referenced by `CIM_MetricForME`.

##### 817 **6.6.2.1.3.2 Pseudo Code**

```
818 &smShowAssociationInstances ("CIM_MetricForME", $instance.getObjectPath() );  
819 &smEnd;
```

### 820 **6.6.2.2 Show Command Form for Multiple Instances – CIM\_BaseMetricValue or** 821 **CIM\_AggregationMetricValue Reference**

822 This command form is when the `show` verb applies to multiple instances. The command form  
823 corresponds to the `show` verb issued against instances of `CIM_MetricForME` where only one reference is  
824 specified and the reference is to an instance of `CIM_BaseMetricValue` or `CIM_AggregationMetricValue`.

#### 825 **6.6.2.2.1 Command Form**

```
826 show <CIM_MetricForME multiple instances>
```

#### 827 **6.6.2.2.2 CIM Requirements**

828 See `CIM_MetricForME` in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of mandatory  
829 properties.

#### 830 **6.6.2.2.3 Behavior Requirements**

##### 831 **6.6.2.2.3.1 Preconditions**

832 `$instance` represents the instance of `CIM_BaseMetricValue` or `CIM_AggregationMetricValue`, which is  
833 referenced by `CIM_MetricForME`.

##### 834 **6.6.2.2.3.2 Pseudo Code**

```
835 &smShowAssociationInstances ("CIM_MetricForME", $instance.getObjectPath() );  
836 &smEnd;
```

837 **6.6.2.3 Show Command Form for a Single Instance – Both References**

838 This command form is when the `show` verb applies to a single instance. This command form corresponds  
 839 to a `show` command issued against `CIM_MetricForME` where both references are specified and therefore  
 840 the desired instance is unambiguously identified.

841 **6.6.2.3.1 Command Form**

```
842 show <CIM_MetricForME single instance>
```

843 **6.6.2.3.2 CIM Requirements**

844 See `CIM_MetricForME` in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of mandatory  
 845 properties.

846 **6.6.2.3.3 Behavior Requirements**847 **6.6.2.3.3.1 Preconditions**

848 `$instanceA` represents the referenced instance of `CIM_BaseMetricValue` or `CIM_AggregationMetricValue`  
 849 through `CIM_MetricForME` association.

850 `$instanceB` represents the other instance of `CIM_ManagedElement` which is referenced by  
 851 `CIM_MetricForME`.

852 **6.6.2.3.3.2 Pseudo Code**

```
853 &smShowAssociationInstance ("CIM_MetricForME", $instanceA.getObjectPath(),  

  854     $instanceB.getObjectPath() );  

  855 &smEnd;
```

856 **6.7 CIM\_ConcreteDependency**

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

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

863 **Table 7 – Command Verb Requirements for CIM\_ConcreteDependency**

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	

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

## 866 6.7.1 Ordering of Results

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

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

## 870 6.7.2 Show

871 This section describes how to implement the `show` verb when applied to an instance of  
872 CIM\_ConcreteDependency. Implementations shall support the use of the `show` verb with  
873 CIM\_ConcreteDependency.

### 874 6.7.2.1 Show Command Form for a Single Instance – CIM\_AggregationMetricValue Reference

875 This command form is used when the `show` verb applies to a single instance. This command form  
876 corresponds to a `show` command issued against instances of CIM\_ConcreteDependency where only one  
877 reference is specified and the reference is to an instance of CIM\_AggregationMetricValue.

#### 878 6.7.2.1.1 Command Form

```
879 show <CIM_ConcreteDependency single instance>
```

#### 880 6.7.2.1.2 CIM Requirements

881 See CIM\_ConcreteDependency in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
882 mandatory properties.

#### 883 6.7.2.1.3 Behavior Requirements

##### 884 6.7.2.1.3.1 Preconditions

885 \$instance represents the instance of CIM\_AggregationMetricValue, which is referenced by  
886 CIM\_ConcreteDependency.

##### 887 6.7.2.1.3.2 Pseudo Code

```
888 &smShowAssociationInstances ( "CIM_ConcreteDependency", $instance.getObjectPath() );  
889 &smEnd;
```

### 890 6.7.2.2 Show Command Form for a Single Instance – CIM\_AggregationMetricDefinition 891 Reference

892 This command form is used when the `show` verb applies to a single instance. This command form  
893 corresponds to a `show` command issued against instances of CIM\_ConcreteDependency where only one  
894 reference is specified and the reference is to an instance of CIM\_AggregationMetricDefinition.

#### 895 6.7.2.2.1 Command Form

```
896 show <CIM_ConcreteDependency single instance>
```

#### 897 6.7.2.2.2 CIM Requirements

898 See CIM\_ConcreteDependency in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
899 mandatory properties.

### 900 6.7.2.2.3 Behavior Requirements

#### 901 6.7.2.2.3.1 Preconditions

902 \$instance represents the instance of CIM\_AggregationMetricDefinition, which is referenced by  
903 CIM\_ConcreteDependency.

#### 904 6.7.2.2.3.2 Pseudo Code

```
905 &smShowAssociationInstances ( "CIM_ConcreteDependency", $instance.GetObjectPath() );  
906 &smEnd;
```

### 907 6.7.2.3 Show Command Form for Multiple Instances – CIM\_BaseMetricValue Reference

908 This command form is when the `show` verb applies to multiple instances. The command form  
909 corresponds to the `show` verb issued against instances of CIM\_ConcreteDependency where only one  
910 reference is specified and the reference is to an instance of CIM\_BaseMetricValue.

#### 911 6.7.2.3.1 Command Form

```
912 show <CIM_ConcreteDependency multiple instances>
```

#### 913 6.7.2.3.2 CIM Requirements

914 See CIM\_ConcreteDependency in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
915 mandatory properties.

### 916 6.7.2.3.3 Behavior Requirements

#### 917 6.7.2.3.3.1 Preconditions

918 \$instance represents the instance of CIM\_BaseMetricValue, which is referenced by  
919 CIM\_ConcreteDependency.

#### 920 6.7.2.3.3.2 Pseudo Code

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

### 923 6.7.2.4 Show Command Form for Multiple Instances – CIM\_BaseMetricDefinition Reference

924 This command form is when the `show` verb applies to multiple instances. The command form  
925 corresponds to the `show` verb issued against instances of CIM\_ConcreteDependency where only one  
926 reference is specified and the reference is to an instance of CIM\_BaseMetricDefinition.

#### 927 6.7.2.4.1 Command Form

```
928 show <CIM_ConcreteDependency multiple instances>
```

#### 929 6.7.2.4.2 CIM Requirements

930 See CIM\_ConcreteDependency in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
931 mandatory properties.

### 932 6.7.2.4.3 Behavior Requirements

#### 933 6.7.2.4.3.1 Preconditions

934 \$instance represents the instance of CIM\_BaseMetricDefinition, which is referenced by  
935 CIM\_ConcreteDependency.

#### 936 6.7.2.4.3.2 Pseudo Code

```
937 &smShowAssociationInstances ("CIM_ConcreteDependency", $instance.getObjectPath() );
938 &smEnd;
```

### 939 6.7.2.5 Show Command Form for a Single Instance – Both References

940 This command form is when the `show` verb applies to a single instance. This command form corresponds  
941 to a `show` command issued against CIM\_ConcreteDependency where both references are specified and  
942 therefore the desired instance is unambiguously identified.

#### 943 6.7.2.5.1 Command Form

```
944 show <CIM_ConcreteDependency single instance>
```

#### 945 6.7.2.5.2 CIM Requirements

946 See CIM\_ConcreteDependency in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
947 mandatory properties.

#### 948 6.7.2.5.3 Behavior Requirements

##### 949 6.7.2.5.3.1 Preconditions

950 \$instanceA represents the referenced instance of CIM\_AggregationMetricValue or  
951 CIM\_AggregationMetricDefinition through CIM\_ConcreteDependency association.

952 \$instanceB represents the other instance of CIM\_BaseMetricValue or CIM\_BaseMetricDefinition which is  
953 referenced by CIM\_ConcreteDependency.

##### 954 6.7.2.5.3.2 Pseudo Code

```
955 &smShowAssociationInstance ("CIM_ConcreteDependency", $instanceA.getObjectPath(),
956     $instanceB.getObjectPath() );
957 &smEnd;
```

## 958 6.8 CIM\_BaseMetricDefinition

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

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

965

Table 8 – Command Verb Requirements for CIM\_BaseMetricDefinition

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	May	See 6.8.2.
set	May	See 6.8.3.
show	Shall	See 6.8.4.
start	May	See 6.8.5.
stop	May	See 6.8.6.

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

### 967 6.8.1 Ordering of Results

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

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

### 971 6.8.2 Reset

972 This section describes how to implement the `reset` verb when applied to an instance of  
973 CIM\_BaseMetricDefinition. Implementations may support the use of the `reset` verb with  
974 CIM\_BaseMetricDefinition.

#### 975 6.8.2.1 General Usage of Reset for a Single Property

##### 976 6.8.2.1.1 Command Form

```
977 reset <CIM_BaseMetricDefinition single instance>
```

##### 978 6.8.2.1.2 CIM Requirements

```
979 uint16 CIM_MetricDefForME.MetricCollectionEnabled;
980 uint32 CIM_MetricService.ControlMetrics (
981     [IN] CIM_ManagedElement REF Subject,
982     [IN] CIM_BaseMetricDefinition REF Definition,
983     [IN] uint16 MetricCollectionEnabled );
```

##### 984 6.8.2.1.3 Behavior Requirements

###### 985 6.8.2.1.3.1 Preconditions

986 \$instance represents the targeted instance of CIM\_BaseMetricDefinition.

```
987 $instance=<CIM_BaseMetricDefinition single instance>
```

###### 988 6.8.2.1.3.2 Pseudo Code

```
989 lControlMetrics ( $instance.getObjectPath(), "Reset" ) ();
990 &smEnd;
```



### 991 **6.8.3 Set**

992 This section describes how to implement the `set` verb when it is applied to an instance of  
 993 `CIM_BaseMetricDefinition`. Implementations may support the use of the `set` verb with  
 994 `CIM_BaseMetricDefinition`.

995 The `set` verb is used to modify descriptive properties of the `CIM_BaseMetricDefinition` instance.

#### 996 **6.8.3.1 General Usage of Set for a Single Property**

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

999 The requirement for supporting modification of a property using this command form shall be equivalent to  
 1000 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined  
 1001 in the [Fan Profile](#).

##### 1002 **6.8.3.1.1 Command Form**

```
1003 set <CIM_BaseMetricDefinition single instance> <propertyname>=<propertyvalue>
```

##### 1004 **6.8.3.1.2 CIM Requirements**

1005 See `CIM_BaseMetricDefinition` in the “CIM Elements” section of the [Fan Profile](#) for the list of mandatory  
 1006 properties.

##### 1007 **6.8.3.1.3 Behavior Requirements**

###### 1008 **6.8.3.1.3.1 Preconditions**

```
1009 $instance=<CIM_BaseMetricDefinition single instance>
```

###### 1010 **6.8.3.1.3.2 Pseudo Code**

```
1011 #propertyName[] = {<propertyname>};  
1012 #propertyValues[] = {<propertyvalue>};  
1013 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );  
1014 &smEnd;
```

#### 1015 **6.8.3.2 General Usage of Set for Multiple Properties**

1016 This command form corresponds to the general usage of the `set` verb to modify multiple properties of a  
 1017 target instance where there is not an explicit relationship between the properties. This is the most  
 1018 common case.

1019 The requirement for supporting modification of a property using this command form shall be equivalent to  
 1020 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined  
 1021 in the [Fan Profile](#).

##### 1022 **6.8.3.2.1 Command Form**

```
1023 set <CIM_BaseMetricDefinition single instance>  
1024 <propertyname1>=<propertyvalue1><propertyname2>=<propertyvalue2>
```

##### 1025 **6.8.3.2.2 CIM Requirements**

1026 See `CIM_BaseMetricDefinition` in the “CIM Elements” section of the [Fan Profile](#) for the list of mandatory  
 1027 properties.

1028 **6.8.3.2.3 Behavior Requirements**1029 **6.8.3.2.3.1 Preconditions**1030 `$instance=<CIM_BaseMetricDefinition single instance>`1031 **6.8.3.2.3.2 Pseudo Code**

```

1032 #propertyName[] = {<propertyname>};
1033 for #i < n
1034 {
1035     #propertyName[#i] = <propertyname#i>
1036     #propertyValue[#i] = <propertyvalue#i>
1037 }
1038 &smSetInstance ( $instance, #propertyName[], #propertyValue[] );
1039 &smEnd;
```

1040 **6.8.4 Show**

1041 This section describes how to implement the `show` verb when applied to an instance of  
 1042 `CIM_BaseMetricDefinition`. Implementations shall support the use of the `show` verb with  
 1043 `CIM_BaseMetricDefinition`.

1044 **6.8.4.1 Show Command Form for Multiple Instances Target**

1045 This command form is used to show many instances of `CIM_BaseMetricDefinition`.

1046 **6.8.4.1.1 Command Form**1047 `show <CIM_BaseMetricDefinition multiple instances>`1048 **6.8.4.1.2 CIM Requirements**

1049 See `CIM_BaseMetricDefinition` in the “CIM Elements” section of the [Fan Profile](#) for the list of mandatory  
 1050 properties.

1051 **6.8.4.1.3 Behavior Requirements**1052 **6.8.4.1.3.1 Preconditions**

1053 `$containerInstance` represents the instance of `CIM_MetricService` which represents the container service  
 1054 and is associated to the targeted instances of `CIM_BaseMetricDefinition` through the  
 1055 `CIM_ServiceAffectsElement` association.

1056 `#all` is true, if the “-all” option was specified with the command; otherwise, `#all` is false.

1057 **6.8.4.1.3.2 Pseudo Code**

```

1058 #Error=smOpAsociators(
1059     $containerInstance->,
1060     "CIM_ServiceAffectsElement",
1061     NULL,
1062     NULL,
1063     NULL,
1064     $definitionInstancePaths[])
1065 if (0 != #Error.code)
```

```

1066 {
1067     &smProcessOpError (#Error);
1068     //includes &smEnd;
1069 }
1070 else
1071 {
1072 for #i < $definitionInstancePaths.length
1073 {
1074 // the class definition for $instance includes two referenced properties,
1075 // MetricCollectionEnabled and RecordedSince.
1076 #Error=smOpReferences(
1077     $definitionInstancePaths->[i],
1078     "CIM_MetricDefForME",
1079     NULL,
1080     NULL,
1081     {"MetricCollectionEnabled","RecordedSince"},
1082     $MDFMEInstancePaths[])
1083 if (0 != #Error.code)
1084 {
1085     &smProcessOpError (#Error);
1086     //includes &smEnd;
1087 }
1088 else
1089 {
1090     #propertynamelist[] = null;
1091     if ( false == #all) {
1092     #propertynamelist[] = <array of mandatory non-key property names (see CIM
1093     Requirements)>;
1094     }
1095 #additionalpropertylist[]={ "MetricCollectionEnabled", "RecordedSince" };
1096 $MDFMEInstance = $MDFMEInstancePaths[1];
1097 $instance.MetricCollectionEnabled = $APMSinstance.MetricCollectionEnabled;
1098 $instance.RecordedSince = $APMSinstance.RecordedSince;
1099 &smShowInstancePseudoProperties(
1100     $instance,
1101     #propertynamelist[],
1102     #additionalpropertylist[] );
1103 }
1104     i++;
1105 }
1106 &smEnd;

```

#### 1107 6.8.4.2 Show Command Form for a Single Instance Target

1108 This command form is used to show a single instance of CIM\_BaseMetricDefinition.

##### 1109 6.8.4.2.1 Command Form

```
1110 show <CIM_BaseMetricDefinition single instance>
```

**1111 6.8.4.2.2 CIM Requirements**

1112 See CIM\_BaseMetricDefinition in the “CIM Elements” section of the [Fan Profile](#) for the list of mandatory  
1113 properties.

**1114 6.8.4.2.3 Behavior Requirements****1115 6.8.4.2.3.1 Preconditions**

1116 \$instance represents the targeted instance of CIM\_BaseMetricDefinition.

```
1117 $instance=<CIM_BaseMetricDefinition single instance>
```

1118 #all is true, if the “-all” option was specified with the command; otherwise, #all is false.

**1119 6.8.4.2.3.2 Pseudo Code**

```
1120 // the class definition for $instance includes two referenced properties,  
1121 // MetricCollectionEnabled and RecordedSince.  
1122 #Error=smOpReferences (  
1123     $instance->,  
1124     "CIM_MetricDefForME",  
1125     NULL,  
1126     NULL,  
1127     {"MetricCollectionEnabled","RecordedSince"},  
1128     $MDFMEInstancePaths[] )  
1129 if ( 0 != #Error.code)  
1130     {  
1131     &smProcessOpError (#Error);  
1132     //includes &smEnd;  
1133     }  
1134 else  
1135     {  
1136     #propertynamelist[] = null;  
1137     if ( false == #all)  
1138         {  
1139         #propertynamelist[] = <array of mandatory non-key property names (see CIM  
1140         Requirements)>;  
1141         }  
1142     #additionalpropertylist[]={ "MetricCollectionEnabled", "RecordedSince" };  
1143     $MDFMEInstance=$MDFMEInstancePaths[1];  
1144     $instance.MetricCollectionEnabled=$APMSinstance.MetricCollectionEnabled;  
1145     $instance.RecordedSince=$APMSinstance.RecordedSince;  
1146     &smShowInstancePseudoProperties (  
1147         $instance,  
1148         #propertynamelist[],  
1149         #additionalpropertylist[] );  
1150     }  
1151 &smEnd;
```

## 1152 6.8.5 Start

1153 This section describes how to implement the `start` verb when applied to an instance of  
 1154 `CIM_BaseMetricDefinition`. Implementations may support the use of the `start` verb with  
 1155 `CIM_BaseMetricDefinition`.

### 1156 6.8.5.1 General Usage of Start for a Single Instance

#### 1157 6.8.5.1.1 Command Form

```
1158 start <CIM_BaseMetricDefinition single instance>
```

#### 1159 6.8.5.1.2 CIM Requirements

```
1160 uint16 CIM_MetricDefForME.MetricCollectionEnabled;
1161 uint32 CIM_MetricService.ControlMetrics(
1162     [IN] CIM_ManagedElement REF Subject,
1163     [IN] CIM_BaseMetricDefinition REF Definition,
1164     [IN] uint16 MetricCollectionEnabled );
```

#### 1165 6.8.5.1.3 Behavior Requirements

##### 1166 6.8.5.1.3.1 Preconditions

1167 `$instance` represents the targeted instance of `CIM_BaseMetricDefinition`.

```
1168 $instance=<CIM_BaseMetricDefinition single instance>
```

##### 1169 6.8.5.1.3.2 Pseudo Code

```
1170 lControlMetrics ( $instance.getObjectPath(), "Enable" ) ();
1171 &smEnd;
```

## 1172 6.8.6 Stop

1173 This section describes how to implement the `stop` verb when applied to an instance of  
 1174 `CIM_BaseMetricDefinition`. Implementations may support the use of the `stop` verb with  
 1175 `CIM_BaseMetricDefinition`.

### 1176 6.8.6.1 General Usage of Stop for a Single Instance

#### 1177 6.8.6.1.1 Command Form

```
1178 stop <CIM_BaseMetricDefinition single instance>
```

#### 1179 6.8.6.1.2 CIM Requirements

```
1180 uint16 CIM_MetricDefForME.MetricCollectionEnabled;
1181 uint32 CIM_MetricService.ControlMetrics(
1182     [IN] CIM_ManagedElement REF Subject,
1183     [IN] CIM_BaseMetricDefinition REF Definition,
1184     [IN] uint16 MetricCollectionEnabled );
```

#### 1185 6.8.6.1.3 Behavior Requirements

##### 1186 6.8.6.1.3.1 Preconditions

1187 `$instance` represents the targeted instance of `CIM_BaseMetricDefinition`.

```
1188 $instance=<CIM_BaseMetricDefinition single instance>
```

1189 **6.8.6.1.3.2 Pseudo Code**

```
1190 lControlMetrics ( $instance.getObjectPath(), "Disable" ) ();
1191 &smEnd;
```

1192 **6.9 CIM\_BaseMetricValue**

1193 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

1194 Table 9 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of  
 1195 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the  
 1196 verb and target. Table 9 is for informational purposes only; in case of a conflict between Table 9 and  
 1197 requirements detailed in the following sections, the text detailed in the following sections supersedes the  
 1198 information in Table 9.

1199 **Table 9 – Command Verb Requirements for CIM\_BaseMetricValue**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.9.2.
show	Shall	See 6.9.3.
start	Not supported	
stop	Not supported	

1200 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,  
 1201 `reset`, `start`, and `stop`.

1202 **6.9.1 Ordering of Results**

1203 When results are returned for multiple instances of `CIM_BaseMetricValue`, implementations shall utilize  
 1204 the following algorithm to produce the natural (that is, default) ordering:

- 1205 • Results for `CIM_BaseMetricValue` are unordered; therefore, no algorithm is defined.

1206 **6.9.2 Set**1207 **6.9.2.1 General Usage of Set for a Single Property**

1208 This command form corresponds to the general usage of the `set` verb to modify a single property of a  
 1209 target instance. The setting of a single property shall be deterministic.

1210 The requirements for supporting modification of a property using this command form shall be equivalent  
 1211 to the requirement for supporting modification of the property using the `ModifyInstance` operation as  
 1212 defined in the [Base Metrics Profile](#).

1213 **6.9.2.1.1 Command Form**

```
1214 set <CIM_BaseMetricValue single instance> <propertyname>=<propertyvalue>
```

### 1215 6.9.2.1.2 CIM Requirements

1216 See CIM\_BaseMetricValue in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
1217 mandatory properties.

### 1218 6.9.2.1.3 Behavior Requirements

#### 1219 6.9.2.1.3.1 Preconditions

```
1220 $instance=< CIM_BaseMetricValue single instance>
```

#### 1221 6.9.2.1.3.2 Pseudo Code

```
1222 #propertyName[] = <propertname>
1223 #propertyValues[] = <propertyvalue>
1224 &smSetInstance ( $instance, #propertyName, #propertyValues );
1225 &smEnd;
```

### 1226 6.9.2.2 General Usage of Set for Multiple Properties

1227 This command form corresponds to the general usage of the set verb to modify multiple properties of a  
1228 target instance. The setting of multiple properties may be deterministic.

1229 The requirements for supporting modification of a property using this command form shall be equivalent  
1230 to the requirement for supporting modification of the property using the ModifyInstance operation as  
1231 defined in the [Base Metrics Profile](#).

#### 1232 6.9.2.2.1 Command Form

```
1233 set <CIM_BaseMetricValue single instance> <propertyname1>=<propertyvalue1>
1234 <propertynameN>=<propertyvalueN>
```

### 1235 6.9.2.2.2 CIM Requirements

1236 See CIM\_BaseMetricValue in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
1237 mandatory properties.

### 1238 6.9.2.2.3 Behavior Requirements

#### 1239 6.9.2.2.3.1 Preconditions

1240 \$instance represents the instance of CIM\_BaseMetricValue.

#### 1241 6.9.2.2.3.2 Pseudo Code

```
1242 for #i < n
1243 {
1244     #propertyName[#i] = <propertname#i>
1245     #propertyValues[#i] = <propertyvalue#i>
1246 }
1247 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
1248 &smEnd;
```

### 1249 6.9.3 Show

1250 The show verb is used to display information about instances of CIM\_BaseMetricValue. Implementations  
1251 shall support the use of the show verb with CIM\_BaseMetricValue.

### 1252 6.9.3.1 Show a Single Instance

1253 This command form is used to display the information about a single instance of CIM\_BaseMetricValue.

#### 1254 6.9.3.1.1 Command Form

```
1255 show <CIM_BaseMetricValue single instance>
```

#### 1256 6.9.3.1.2 CIM Requirements

1257 See CIM\_BaseMetricValue in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
1258 mandatory properties.

#### 1259 6.9.3.1.3 Behavior Requirements

##### 1260 6.9.3.1.3.1 Preconditions

1261 \$instance represents the instance of CIM\_BaseMetricValue.

1262 #all is true, if the “-all” option was specified with the command; otherwise, #all is false.

1263 #propertylist[] is an array of mandatory non-key property names.

##### 1264 6.9.3.1.3.2 Pseudo Code

```
1265 if (false != #all) { #propertylist[] = NULL; }  
1266 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );  
1267 &smEnd;
```

### 1268 6.9.3.2 Show Multiple Instances

1269 This command form is used to display the information about multiple instances of CIM\_BaseMetricValue.  
1270 This command form corresponds to UFsT-based selection within a scoping system.

#### 1271 6.9.3.2.1 Command Form

```
1272 show <CIM_BaseMetricValue multiple instances>
```

#### 1273 6.9.3.2.2 CIM Requirements

1274 See CIM\_BaseMetricValue in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
1275 mandatory properties.

#### 1276 6.9.3.2.3 Behavior Requirements

##### 1277 6.9.3.2.3.1 Preconditions

1278 \$containerInstance represents the instance of CIM\_BaseMetricDefinition to which the instance of  
1279 CIM\_BaseMetricValue being displayed is scoped. The CIM\_BaseMetricDefinition is associated to  
1280 targeted instances of CIM\_BaseMetricValue via a CIM\_MetricInstance association.

1281 #all is true, if the “-all” option was specified with the command; otherwise, #all is false.

1282 #propertylist[] is an array of mandatory non-key property names.



1283 **6.9.3.2.3.2 Pseudo Code**

```

1284 if (false != #all) { #propertylist[] = NULL; }
1285 &smShowInstances ( "CIM_BaseMetricValue", "CIM_MetricInstance",
1286     $containerInstance.getObjectPath(), #propertylist[] );
1287 &smEnd;

```

1288 **6.10 CIM\_AggregationMetricDefinition**

1289 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

1290 Table 10 lists each SM CLP verb, the required level of support for the verb in conjunction with instances  
 1291 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the  
 1292 verb and target. Table 10 is for informational purposes only; in case of a conflict between Table 10 and  
 1293 requirements detailed in the following sections, the text detailed in the following sections supersedes the  
 1294 information in Table 10.

1295 **Table 10 – Command Verb Requirements for CIM\_AggregationMetricDefinition**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	May	See 6.10.2.
set	May	See 6.10.3.
show	Shall	See 6.10.4.
start	May	See 6.10.5.
stop	May	See 6.10.6.

1296 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, and `load`.

1297 **6.10.1 Ordering of Results**

1298 When results are returned for multiple instances of `CIM_AggregationMetricDefinition`, implementations  
 1299 shall utilize the following algorithm to produce the natural (that is, default) ordering:

- 1300 • Results for `CIM_AggregationMetricDefinition` are unordered; therefore, no algorithm is defined.

1301 **6.10.2 Reset**

1302 This section describes how to implement the `reset` verb when applied to an instance of  
 1303 `CIM_AggregationMetricDefinition`. Implementations may support the use of the `reset` verb with  
 1304 `CIM_AggregationMetricDefinition`.

1305 **6.10.2.1 General Usage of Reset for a Single Property**1306 **6.10.2.1.1 Command Form**

```

1307 reset <CIM_AggregationMetricDefinition single instance>

```

1308 **6.10.2.1.2 CIM Requirements**

```

1309 uint16 CIM_MetricDefForME.MetricCollectionEnabled;
1310 uint32 CIM_MetricService.ControlMetrics(
1311     [IN] CIM_ManagedElement REF Subject,
1312     [IN] CIM_AggregationMetricDefinition REF Definition,
1313     [IN] uint16 MetricCollectionEnabled );

```

1314 **6.10.2.1.3 Behavior Requirements**1315 **6.10.2.1.3.1 Preconditions**

1316 \$instance represents the targeted instance of CIM\_AggregationMetricDefinition.

```

1317 $instance=<CIM_AggregationMetricDefinition single instance>

```

1318 **6.10.2.1.3.2 Pseudo Code**

```

1319 lControlMetrics ( $instance.getObjectPath(), "Reset" ) ();
1320 &smEnd;

```

1321 **6.10.3 Set**

1322 This section describes how to implement the `set` verb when it is applied to an instance of  
 1323 CIM\_AggregationMetricDefinition. Implementations may support the use of the `set` verb with  
 1324 CIM\_AggregationMetricDefinition.

1325 The `set` verb is used to modify descriptive properties of the CIM\_AggregationMetricDefinition instance.

1326 **6.10.3.1 General Usage of Set for a Single Property**

1327 This command form corresponds to the general usage of the `set` verb to modify a single property of a  
 1328 target instance. This is the most common case.

1329 The requirement for supporting modification of a property using this command form shall be equivalent to  
 1330 the requirement for supporting modification of the property using the ModifyInstance operation as defined  
 1331 in the [Fan Profile](#).

1332 **6.10.3.1.1 Command Form**

```

1333 set <CIM_AggregationMetricDefinition single instance> <propertyname>=<propertyvalue>

```

1334 **6.10.3.1.2 CIM Requirements**

1335 See CIM\_AggregationMetricDefinition in the “CIM Elements” section of the [Fan Profile](#) for the list of  
 1336 mandatory properties.

1337 **6.10.3.1.3 Behavior Requirements**1338 **6.10.3.1.3.1 Preconditions**

```

1339 $instance=<CIM_AggregationMetricDefinition single instance>

```

1340 **6.10.3.1.3.2 Pseudo Code**

```

1341 #propertyName[] = {<propertyname>};
1342 #propertyValues[] = {<propertyvalue>};
1343 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
1344 &smEnd;

```

### 1345 6.10.3.2 General Usage of Set for Multiple Properties

1346 This command form corresponds to the general usage of the `set` verb to modify multiple properties of a  
1347 target instance where there is not an explicit relationship between the properties. This is the most  
1348 common case.

1349 The requirement for supporting modification of a property using this command form shall be equivalent to  
1350 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined  
1351 in the [Fan Profile](#).

#### 1352 6.10.3.2.1 Command Form

```
1353 set <CIM_AggregationMetricDefinition single instance> <propertyname1>=<propertyvalue1>  
1354 <propertynamen>=<propertyvaluen>
```

#### 1355 6.10.3.2.2 CIM Requirements

1356 See `CIM_AggregationMetricDefinition` in the “CIM Elements” section of the [Fan Profile](#) for the list of  
1357 mandatory properties.

#### 1358 6.10.3.2.3 Behavior Requirements

##### 1359 6.10.3.2.3.1 Preconditions

```
1360 $instance=<CIM_AggregationMetricDefinition single instance>
```

##### 1361 6.10.3.2.3.2 Pseudo Code

```
1362 #propertyName[] = {<propertyname>};  
1363 for #i < n  
1364 {  
1365     #propertyName[#i] = <propertyname#i>  
1366     #propertyValue[#i] = <propertyvalue#i>  
1367 }  
1368 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );  
1369 &smEnd;
```

### 1370 6.10.4 Show

1371 This section describes how to implement the `show` verb when applied to an instance of  
1372 `CIM_AggregationMetricDefinition`. Implementations shall support the use of the `show` verb with  
1373 `CIM_AggregationMetricDefinition`.

#### 1374 6.10.4.1 Show Command Form for Multiple Instances Target

1375 This command form is used to show many instances of `CIM_AggregationMetricDefinition`.

##### 1376 6.10.4.1.1 Command Form

```
1377 show <CIM_AggregationMetricDefinition multiple instances>
```

##### 1378 6.10.4.1.2 CIM Requirements

1379 See `CIM_AggregationMetricDefinition` in the “CIM Elements” section of the [Fan Profile](#) for the list of  
1380 mandatory properties.

1381 **6.10.4.1.3 Behavior Requirements**1382 **6.10.4.1.3.1 Preconditions**

1383 \$containerInstance represents the instance of CIM\_MetricService which represents the container service  
 1384 and is associated to the targeted instances of CIM\_AggregationMetricDefinition through the  
 1385 CIM\_ServiceAffectsElement association.

1386 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

1387 **6.10.4.1.3.2 Pseudo Code**

```

1388 #Error=smOpAsociators (
1389     $containerinstance->,
1390     "CIM_ServiceAffectsElement",
1391     NULL,
1392     NULL,
1393     NULL,
1394     $definitionInstancePaths[] )
1395 if (0 != #Error.code)
1396 {
1397     &smProcessOpError (#Error);
1398     //includes &smEnd;
1399 }
1400 else
1401 {
1402     for #i < $definitionInstancePaths.length
1403     {
1404         // the class definition for $instance includes two referenced properties,
1405         // MetricCollectionEnabled and RecordedSince.
1406         #Error=smOpReferences(
1407             $definitionInstancePaths->[i],
1408             "CIM_MetricDefForME",
1409             NULL,
1410             NULL,
1411             {"MetricCollectionEnabled","RecordedSince"},
1412             $MDFMEInstancePaths[])
1413         if (0 != #Error.code)
1414         {
1415             &smProcessOpError (#Error);
1416             //includes &smEnd;
1417         }
1418         else
1419         {
1420             #propertynamelist[] = null;
1421             if ( false == #all) {
1422                 #propertynamelist[] = <array of mandatory non-key property names (see CIM
1423                 Requirements)>;
1424             }
1425             #additionalpropertylist[]={ "MetricCollectionEnabled", "RecordedSince" };
1426             $MDFMEInstance = $MDFMEInstancePaths[1];

```

```

1427     $instance.MetricCollectionEnabled
1428     =$APMSinstance.MetricCollectionEnabled;
1429     $instance.RecordedSince =$APMSinstance.RecordedSince;
1430     &smShowInstancePseudoProperties(
1431         $instance,
1432         #propertynamelist[],
1433         #additionalpropertylist[] );
1434     }
1435     i++;
1436 }
1437 &smEnd;

```

#### 1438 6.10.4.2 Show Command Form for a Single Instance Target

1439 This command form is used to show a single instance of CIM\_AggregationMetricDefinition.

##### 1440 6.10.4.2.1 Command Form

```
1441 show <CIM_AggregationMetricDefinition single instance>
```

##### 1442 6.10.4.2.2 CIM Requirements

1443 See CIM\_AggregationMetricDefinition in the “CIM Elements” section of the [Fan Profile](#) for the list of  
1444 mandatory properties.

##### 1445 6.10.4.2.3 Behavior Requirements

###### 1446 6.10.4.2.3.1 Preconditions

1447 \$instance represents the targeted instance of CIM\_AggregationMetricDefinition.

```
1448 $instance=<CIM_AggregationMetricDefinition single instance>
```

1449 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

###### 1450 6.10.4.2.3.2 Pseudo Code

```

1451 // the class definition for $instance includes two referenced properties,
1452 // MetricCollectionEnabled and RecordedSince.
1453 #Error=smOpReferences (
1454     $instance->,
1455     "CIM_MetricDefForME",
1456     NULL,
1457     NULL,
1458     {"MetricCollectionEnabled","RecordedSince"},
1459     $MDFMEInstancePaths[] )
1460 if (0 != #Error.code)
1461 {
1462     &smProcessOpError (#Error);
1463     //includes &smEnd;
1464 }
1465 else
1466 {
1467     #propertynamelist[] = null;
1468     if ( false == #all)

```

```

1469     {
1470         #propertynamelist[] = <array of mandatory non-key property names (see CIM
1471             Requirements)>;
1472     }
1473     #additionalpropertylist[]={“MetricCollectionEnabled”, “RecordedSince”};
1474     $MDFMEInstance=$MDFMEInstancePaths[1];
1475     $instance.MetricCollectionEnabled=$APMSinstance.MetricCollectionEnabled;
1476     $instance.RecordedSince=$APMSinstance.RecordedSince;
1477     &smShowInstancePseudoProperties(
1478         $instance,
1479         #propertynamelist[],
1480         #additionalpropertylist[]);
1481     }
1482     &smEnd;

```

## 1483 6.10.5 Start

### 1484 6.10.5.1 General Usage of Start for a Single Property

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

#### 1488 6.10.5.1.1 Command Form

```
1489 start <CIM_AggregationMetricDefinition single instance>
```

#### 1490 6.10.5.1.2 CIM Requirements

```

1491 uint16 CIM_MetricDefForME.MetricCollectionEnabled;
1492 uint32 CIM_MetricService.ControlMetrics(
1493     [IN] CIM_ManagedElement REF Subject,
1494     [IN] CIM_BaseMetricDefinition REF Definition,
1495     [IN] uint16 MetricCollectionEnabled );

```

#### 1496 6.10.5.1.3 Behavior Requirements

##### 1497 6.10.5.1.3.1 Preconditions

1498 `$instance` represents the targeted instance of `CIM_AggregationMetricDefinition`.

```
1499 $instance=<CIM_AggregationMetricDefinition single instance>
```

##### 1500 6.10.5.1.3.2 Pseudo Code

```

1501 lControlMetrics ( $instance.getObjectPath(), “Enable” ) ();
1502 &smEnd;

```

## 1503 6.10.6 Stop

### 1504 6.10.6.1 General Usage of Stop for a Single Property

1505 This section describes how to implement the `stop` verb when applied to an instance of  
 1506 `CIM_AggregationMetricDefinition`. Implementations may support the use of the `stop` verb with  
 1507 `CIM_AggregationMetricDefinition`.

1508 **6.10.6.1.1 Command Form**

1509 `stop <CIM_AggregationMetricDefinition single instance>`

1510 **6.10.6.1.2 CIM Requirements**

```
1511 uint16 CIM_MetricDefForME.MetricCollectionEnabled;
1512 uint32 CIM_MetricService.ControlMetrics(
1513     [IN] CIM_ManagedElement REF Subject,
1514     [IN] CIM_BaseMetricDefinition REF Definition,
1515     [IN] uint16 MetricCollectionEnabled );
```

1516 **6.10.6.1.3 Behavior Requirements**

1517 **6.10.6.1.3.1 Preconditions**

1518 \$instance represents the targeted instance of CIM\_AggregationMetricDefinition.

1519 `$instance=<CIM_AggregationMetricDefinition single instance>`

1520 **6.10.6.1.3.2 Pseudo Code**

```
1521 lControlMetrics ( $instance.getObjectPath(), "Disable" ) ();
1522 &smEnd;
```

1523 **6.11 CIM\_AggregationMetricValue**

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

1525 Table 11 lists each SM CLP verb, the required level of support for the verb in conjunction with instances  
 1526 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the  
 1527 verb and target. Table 11 is for informational purposes only; in case of a conflict between Table 11 and  
 1528 requirements detailed in the following sections, the text detailed in the following sections supersedes the  
 1529 information in Table 11.

1530 **Table 11 – Command Verb Requirements for CIM\_AggregationMetricValue**

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	

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

### 1533 6.11.1 Ordering of Results

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

- 1536 • Results for CIM\_AggregationMetricValue are unordered; therefore, no algorithm is defined.

### 1537 6.11.2 Set

#### 1538 6.11.2.1 General Usage of Set for a Single Property

1539 This command form corresponds to the general usage of the set verb to modify a single property of a  
1540 target instance. The setting of a single property shall be deterministic.

1541 The requirements for supporting modification of a property using this command form shall be equivalent  
1542 to the requirement for supporting modification of the property using the ModifyInstance operation as  
1543 defined in the [Base Metrics Profile](#).

##### 1544 6.11.2.1.1 Command Form

```
1545 set <CIM_AggregationMetricValue single instance> <propertyname>=<propertyvalue>
```

##### 1546 6.11.2.1.2 CIM Requirements

1547 See CIM\_AggregationMetricValue in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
1548 mandatory properties.

##### 1549 6.11.2.1.3 Behavior Requirements

###### 1550 6.11.2.1.3.1 Preconditions

```
1551 $instance=<CIM_AggregationMetricValue single instance>
```

###### 1552 6.11.2.1.3.2 Pseudo Code

```
1553 #propertyNames[] = <propertyname>  
1554 #propertyValues[] = <propertyvalue>  
1555 &smSetInstance ( $instance, #propertyNames, #propertyValues );  
1556 &smEnd;
```

#### 1557 6.11.2.2 General Usage of Set for Multiple Properties

1558 This command form corresponds to the general usage of the set verb to modify multiple properties of a  
1559 target instance. The setting of multiple properties may be deterministic.

1560 The requirements for supporting modification of a property using this command form shall be equivalent  
1561 to the requirement for supporting modification of the property using the ModifyInstance operation as  
1562 defined in the [Base Metrics Profile](#).

##### 1563 6.11.2.2.1 Command Form

```
1564 set <CIM_AggregationMetricValue single instance> <propertyname1>=<propertyvalue1>  
1565 <propertynameN>=<propertyvalueN>
```

##### 1566 6.11.2.2.2 CIM Requirements

1567 See CIM\_AggregationMetricValue in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
1568 mandatory properties.



### 1569 6.11.2.2.3 Behavior Requirements

#### 1570 6.11.2.2.3.1 Preconditions

1571 \$instance represents the instance of CIM\_AggregationMetricValue.

#### 1572 6.11.2.2.3.2 Pseudo Code

```

1573 for #i < n
1574 {
1575     #propertyName[#i] = <propertyname#i>
1576     #propertyValues[#i] = <propertyvalue#i>
1577 }
1578 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
1579 &smEnd;
```

### 1580 6.11.3 Show

1581 The `show` verb is used to display information about instances of CIM\_AggregationMetricValue.  
 1582 Implementations shall support the use of the `show` verb with CIM\_AggregationMetricValue.

#### 1583 6.11.3.1 Show a Single Instance

1584 This command form is used to display the information about a single instance of  
 1585 CIM\_AggregationMetricValue.

##### 1586 6.11.3.1.1 Command Form

```
1587 show <CIM_AggregationMetricValue single instance>
```

##### 1588 6.11.3.1.2 CIM Requirements

1589 See CIM\_AggregationMetricValue in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
 1590 mandatory properties.

#### 1591 6.11.3.1.3 Behavior Requirements

##### 1592 6.11.3.1.3.1 Preconditions

1593 \$instance represents the instance of CIM\_AggregationMetricValue.

1594 #all is true, if the “-all” option was specified with the command; otherwise, #all is false.

1595 #propertylist[] is an array of mandatory non-key property names.

##### 1596 6.11.3.1.3.2 Pseudo Code

```

1597 if (false != #all) { #propertylist[] = NULL; }
1598 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
1599 &smEnd;
```

#### 1600 6.11.3.2 Show Multiple Instances

1601 This command form is used to display the information about multiple instances of  
 1602 CIM\_AggregationMetricValue. This command form corresponds to UFsT-based selection within a scoping  
 1603 system.

1604 **6.11.3.2.1 Command Form**1605 `show <CIM_AggregationMetricValue multiple instances>`1606 **6.11.3.2.2 CIM Requirements**1607 See CIM\_AggregationMetricValue in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
1608 mandatory properties.1609 **6.11.3.2.3 Behavior Requirements**1610 **6.11.3.2.3.1 Preconditions**1611 \$containerInstance represents the instance of CIM\_AggregationMetricDefinition to which the instance of  
1612 CIM\_AggregationMetricValue being displayed is scoped. The CIM\_AggregationMetricDefinition is  
1613 associated to targeted instances of CIM\_AggregationMetricValue via a CIM\_MetricInstance association.

1614 #all is true, if the “-all” option was specified with the command; otherwise, #all is false.

1615 #propertylist[] is an array of mandatory non-key property names.

1616 **6.11.3.2.3.2 Pseudo Code**1617 

```
if (false != #all) { #propertylist[] = NULL; }
1618 &smShowInstances ( "CIM_AggregationMetricValue", "CIM_MetricInstance",
1619     $containerInstance.getObjectPath(), #propertylist[] );
1620 &smEnd;
```

1621 **6.12 CIM\_MetricServiceCapabilities**1622 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).1623 Table 12 lists each SM CLP verb, the required level of support for the verb in conjunction with instances  
1624 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the  
1625 verb and target. Table 12 is for informational purposes only; in case of a conflict between Table 12 and  
1626 requirements detailed in the following sections, the text detailed in the following sections supersedes the  
1627 information in Table 12.1628 **Table 12 – Command Verb Requirements for CIM\_MetricServiceCapabilities**

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	

1629 No mapping is defined for the following verbs for the specified target: create, delete, dump, load,  
1630 reset, set, start, and stop.

## 1631 6.12.1 Ordering of Results

1632 When results are returned for multiple instances of CIM\_MetricServiceCapabilities, implementations shall  
1633 utilize the following algorithm to produce the natural (that is, default) ordering:

- 1634 • Results for CIM\_MetricServiceCapabilities are unordered; therefore, no algorithm is defined.

## 1635 6.12.2 Show

1636 This section describes how to implement the `show` verb when applied to an instance of  
1637 CIM\_MetricServiceCapabilities. Implementations shall support the use of the `show` verb with  
1638 CIM\_MetricServiceCapabilities.

1639 The `show` verb is used to display information about an instance or instances of the  
1640 CIM\_MetricServiceCapabilities class.

### 1641 6.12.2.1 Show a Single Instance

1642 This command form is for the `show` verb applied to a single instance of CIM\_MetricServiceCapabilities.

#### 1643 6.12.2.1.1 Command Form

```
1644 show <CIM_MetricServiceCapabilities single instance>
```

#### 1645 6.12.2.1.2 CIM Requirements

1646 See CIM\_MetricServiceCapabilities in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
1647 mandatory properties.

#### 1648 6.12.2.1.3 Behavior Requirements

##### 1649 6.12.2.1.3.1 Preconditions

1650 #all is true, if the “-all” option was specified with the command; otherwise, #all is false.

```
1651 $instance=<CIM_MetricServiceCapabilities single instance>
```

##### 1652 6.12.2.1.3.2 Pseudo Code

```
1653 #propertylist[] = NULL;
1654 if ( false == #all )
1655 {
1656     #propertylist[] = { //all mandatory non-key properties }
1657 }
1658 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
1659 &smEnd;
```

### 1660 6.12.2.2 Show Multiple Instances

1661 This command form is for the `show` verb applied to multiple instances of CIM\_MetricServiceCapabilities.  
1662 This command form corresponds to UFsT-based selection within a capabilities collection.

#### 1663 6.12.2.2.1 Command Form

```
1664 show <CIM_MetricServiceCapabilities multiple instances>
```

1665 **6.12.2.2.2 CIM Requirements**

1666 See CIM\_MetricServiceCapabilities in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of  
1667 mandatory properties.

1668 **6.12.2.2.3 Behavior Requirements**1669 **6.12.2.2.3.1 Preconditions**

1670 \$containerInstance represents the instance of CIM\_ConcreteCollection with ElementName property that  
1671 contains “Capabilities” and is associated to the targeted instances of CIM\_MetricServiceCapabilities  
1672 through the CIM\_MemberOfCollection association.

1673 #all is true, if the “-all” option was specified with the command; otherwise, #all is false.

1674 **6.12.2.2.3.2 Pseudo Code**

```
1675 #propertylist[] = NULL;
1676 if ( false == #all )
1677     {
1678         #propertylist[] = { //all mandatory non-key properties }
1679     }
1680 &smShowInstances ( "CIM_MetricServiceCapabilities", "CIM_MemberOfCollection",
1681     $containerInstance.getObjectPath(), #propertylist[] );
1682 &smEnd;
```

1683 **6.13 CIM\_MetricService**

1684 The cd and help verbs shall be supported as described in [DSP0216](#).

1685 Table 13 lists each SM CLP verb, the required level of support for the verb in conjunction with instances  
1686 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the  
1687 verb and target. Table 13 is for informational purposes only; in case of a conflict between Table 13 and  
1688 requirements detailed in the following sections, the text detailed in the following sections supersedes the  
1689 information in Table 13.

1690 **Table 13 – Command Verb Requirements for CIM\_MetricService**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.13.2.
show	Shall	See 6.13.3.
start	Not supported	
stop	Not supported	

1691 No mapping is defined for the following verbs for the specified target: create, delete, dump, load,  
1692 reset, start, and stop.

### 1693 6.13.1 Ordering of Results

1694 When results are returned for multiple instances of CIM\_MetricService, implementations shall utilize the  
1695 following algorithm to produce the natural (that is, default) ordering:

- 1696 • Results for CIM\_MetricService are unordered; therefore, no algorithm is defined.

### 1697 6.13.2 Set

1698 This section describes how to implement the `set` verb when it is applied to an instance of  
1699 CIM\_MetricService. The `set` verb is used to set properties on an instance of CIM\_MetricService.

1700 Implementations may support the use of the `set` verb with CIM\_MetricService.

#### 1701 6.13.2.1 General Usage of Set for a Single Property

1702 This command form corresponds to the general usage of the `set` verb to modify a single property of a  
1703 target instance. The setting of a single property shall be deterministic.

1704 The requirements for supporting modification of a property using this command form shall be equivalent  
1705 to the requirement for supporting modification of the property using the ModifyInstance operation as  
1706 defined in the [Base Metrics Profile](#).

##### 1707 6.13.2.1.1 Command Form

```
1708 set <CIM_MetricService single instance> <propertyname>=<propertyvalue>
```

##### 1709 6.13.2.1.2 CIM Requirements

1710 See CIM\_MetricService in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of mandatory  
1711 properties.

##### 1712 6.13.2.1.3 Behavior Requirements

###### 1713 6.13.2.1.3.1 Preconditions

```
1714 $instance=<CIM_MetricService single instance>
```

###### 1715 6.13.2.1.3.2 Pseudo Code

```
1716 #propertyName[] = <propertyname>
1717 #propertyValues[] = <propertyvalue>
1718 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
1719 &smEnd;
```

### 1720 6.13.2.2 General Usage of Set for Multiple Properties

1721 This command form corresponds to the general usage of the `set` verb to modify multiple properties of a  
1722 target instance. The setting of multiple properties may be deterministic.

1723 The requirements for supporting modification of a property using this command form shall be equivalent  
1724 to the requirement for supporting modification of the property using the ModifyInstance operation as  
1725 defined in the [Base Metrics Profile](#).

#### 1726 6.13.2.2.1 Command Form

```
1727 set <CIM_MetricService single instance> <propertyname1>=<propertyvalue1>
1728 <propertynameN>=<propertyvalueN>
```

1729 **6.13.2.2.2 CIM Requirements**

1730 See CIM\_MetricService in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of mandatory  
1731 properties.

1732 **6.13.2.2.3 Behavior Requirements**1733 **6.13.2.2.3.1 Preconditions**

1734 \$instance represents the instance of CIM\_MetricService.

1735 **6.13.2.2.3.2 Pseudo Code**

```
1736 for #i < n
1737 {
1738     #propertyName[#i] = <propertyname#i>
1739     #propertyValues[#i] = <propertyvalue#i>
1740 }
1741 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
1742 &smEnd;
```

1743 **6.13.3 Show**

1744 The `show` verb is used to display information about instances of CIM\_MetricService. Implementations  
1745 shall support the use of the `show` verb with CIM\_MetricService.

1746 **6.13.3.1 Show Command Form for a Single Instance**

1747 This command form is used to show a single instance of CIM\_MetricService.

1748 **6.13.3.1.1 Command Form**

```
1749 show <CIM_MetricService single instance>
```

1750 **6.13.3.1.2 CIM Requirements**

1751 See CIM\_MetricService in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of mandatory  
1752 properties.

1753 **6.13.3.1.3 Behavior Requirements**1754 **6.13.3.1.3.1 Preconditions**

1755 \$instance represents the instance of CIM\_MetricService.

1756 #all is true, if the “-all” option was specified with the command; otherwise, #all is false.

1757 #propertylist[] is an array of mandatory non-key property names.

1758 **6.13.3.1.3.2 Pseudo Code**

```
1759 if (false != #all) { #propertylist[] = NULL; }
1760 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
1761 &smEnd;
```

### 1762 6.13.3.2 Show Command Form for Multiple Instances

1763 This command form is used to show multiple instances of CIM\_MetricService.

#### 1764 6.13.3.2.1 Command Form

```
1765 show <CIM_MetricService multiple instances>
```

#### 1766 6.13.3.2.2 CIM Requirements

1767 See CIM\_MetricService in the “CIM Elements” section of the [Base Metrics Profile](#) for the list of mandatory  
1768 properties.

#### 1769 6.13.3.2.3 Behavior Requirements

##### 1770 6.13.3.2.3.1 Preconditions

1771 \$containerInstance contains the instance of CIM\_ComputerSystem that is associated to the targeted  
1772 instances of CIM\_MetricService through the CIM\_HostedService association.

1773 #all is true, if the “-all” option was specified with the command; otherwise, #all is false.

##### 1774 6.13.3.2.3.2 Pseudo Code

```
1775 #propertylist[] = NULL;  
1776 if ( false == #all )  
1777 {  
1778     #propertylist[] = {<array of mandatory non-key property names (see CIM  
1779     Requirements)>}  
1780 }  
1781 &smShowInstances ( "CIM_MetricService", "CIM_HostedService",  
1782     $containerInstance.getObjectPath(), #propertylist[] );  
1783 &smEnd;
```

1784  
1785  
1786  
1787  
1788

# ANNEX A

(informative)

## Change Log

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

1789