**Document Number: DSP0845**

**Date: 2009-06-04**

**Version: 1.0.0**

# Base Metrics Profile SM CLP Command Mapping Specification

**Document Type: Specification**

**Document Status: DMTF Standard**

**Document Language: E**

# CONTENTS

## Tables

# Foreword

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

## Conventions

The pseudo-code conventions utilized in this document are the Recipe Conventions as defined in SNIA SMI-S 1.1.0, section 7.6.

## Acknowledgements

- Khachatur Papanyan – Dell

# Introduction

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

The target audience for this specification is implementers of the SM CLP support for the *Base Metrics Profile*.

# Base Metrics Profile SM CLP Command Mapping Specification

## 1 Scope

This specification contains the requirements for an implementation of the SM CLP to provide access to, and implement the behaviors of, the *Base Metrics Profile*.

## 2 Normative References

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

### 2.1 Approved References

DMTF DSP1053, *Base Metrics Profile 1.0*, http://www.dmtf.org/standards/published_documents/DSP1053_1.0.pdf

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

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

### 2.1 Other References

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

## 3 Terms and Definitions

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

**3.1**
**can**
used for statements of possibility and capability, whether material, physical, or causal

**3.2**
**cannot**
used for statements of possibility and capability, whether material, physical or causal

**3.3**
**conditional**
indicates requirements to be followed strictly in order to conform to the document when the specified conditions are met

**3.4**

**mandatory**

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

**3.5**

**may**

indicates a course of action permissible within the limits of the document

**3.6**

**need not**

indicates a course of action permissible within the limits of the document

**3.7**

**optional**

indicates a course of action permissible within the limits of the document

**3.8**

**shall**

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

**3.9**

**shall not**

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

**3.10**

**should**

indicates that among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others, or that a certain course of action is preferred but not necessarily required

**3.11**

**should not**

indicates that a certain possibility or course of action is deprecated but not prohibited

# 4 Symbols and Abbreviated Terms

The following symbols and abbreviations are used in this document.

**4.1**

**CIM**

Common Information Model

**4.2**

**CLP**

Command Line Protocol

**4.3**

**DMTF**

Distributed Management Task Force

**4.4**
**SM**
Server Management

**4.5**
**SMI-S**
Storage Management Initiative Specification

**4.6**
**SNIA**
Storage Networking Industry Association

**4.7**
**UFsT**
User Friendly selectionTag

# 5 Recipes

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

- smShowInstance( )
- smShowInstances( )
- smSetInstance( )
- smShowAssocationInstance( )
- smShowAssociationInstances( )
- smRequestStateChange( )
- smStartRSC()
- smStopRSC( )

## 5.1 Local Recipes

### 5.1.1 lControlMetrics

This function implements an invocation of the ControlMetrics( ) method on CIM_MetricService. The function handles the implementation through the production of Command Status.

```
sub void lControlMetrics($metricDef->, string #requestedState) {
//find associated CIM_MetricServic instance and store the pointer in $service
   #Error = &smOpAssociators ($metricDef->, “CIM_ServiceAffectsElement”,
   “Dell_OEMPowerUtilizationManagementService”, “AffectedElement”,
   “AffectingElement”, NULL, $outInstancePaths->[]);
   if (0 != #Error.code)
   {
      &smProcessOpError (#Error);
      //includes &smEnd;
   }
   if ( 0 < $outInstancePaths.length ) {
      $service = $outInstancePaths->[0];
   }
```

```
else   {
   // generic failure
   $OperationError = smNewInstance("CIM_Error");
   //CIM_ERR_FAILED
   $OperationError.CIMStatusCode = 1;
   //Other
   $OperationError.ErrorType = 1;
   //Low
   $OperationError.PerceivedSeverity = 2;
   $OperationError.OwningEntity = DMTF:SMCLP;
   $OperationError.MessageID = 0x00000002;
   $OperationError.Message = "Failed. No further information is available.";
   &smAddError($job, $OperationError);
   &smMakeCommandStatus($job);
   &smEnd;
}
#intRequestedState = <integer value of valuemap string #requestedState>
%InArguments[] = {newArgument("Subject", NULL),
       newArgument("Definition", $metricDef->),
       newArgument("MetricCollectionEnabled", #intRequestedState) };
%OutArguments[] = {};
#Error = smOpInvokeMethod ($service->,
       "ControlMetrics",
       %InArguments[],
       %OutArguments[],
       #returnStatus);
if (0 != #Error.code)  {
   //method invocation failed
   if ( (null != #Error.$error) && (null != #Error.$error[0]) )    {
      //if the method invocation contains an embedded error
    //use it for the Error for the overall job
    &smAddError($job, #Error.$error[0]);
    &smMakeCommandStatus($job);
    &smEnd;
   }
else if (#Error.code == 17)   {
//trap for CIM_METHOD_NOT_FOUND
//and make nice Unsupported msg.
   //unsupported
   $OperationError = smNewInstance("CIM_Error");
   //CIM_ERR_NOT_SUPPORTED
   $OperationError.CIMStatusCode = 7;
   //Other
   $OperationError.ErrorType = 1;
   //Low
   $OperationError.PerceivedSeverity = 2;
   $OperationError.OwningEntity = DMTF:SMCLP;
   $OperationError.MessageID = 0x00000001;
   $OperationError.Message = "Operation is not supported.";
```

```
        &smAddError($job, $OperationError);
        &smMakeCommandStatus($job);
        &smEnd;
    }
    else {
        //operation failed, but no detailed error instance, need to make one up
        //make an Error instance and associate with job for Operation
            $OperationError = smNewInstance(“CIM_Error”);
            //CIM_ERR_FAILED
            $OperationError.CIMStatusCode = 1;
            //Software Error
            $OperationError.ErrorType = 4;
            //Unknown
            $OperationError.PerceivedSeverity = 0;
            $OperationError.OwningEntity = DMTF:SMCLP;
            $OperationError.MessageID = 0x00000009;
            $OperationError.Message = “An internal software error has occurred.”;
            &smAddError($job, $OperationError);
            &smMakeCommandStatus($job);
            &smEnd;
    }
    else {
        //operation failed, but no detailed error instance, need to make one up
        //make an Error instance and associate with job for Operation
            $OperationError = smNewInstance(“CIM_Error”);
            //CIM_ERR_FAILED
            $OperationError.CIMStatusCode = 1;
            //Software Error
            $OperationError.ErrorType = 4;
            //Unknown
            $OperationError.PerceivedSeverity = 0;
            $OperationError.OwningEntity = DMTF:SMCLP;
            $OperationError.MessageID = 0x00000009;
            $OperationError.Message = “An internal software error has occurred.”;
            &smAddError($job, $OperationError);
            &smMakeCommandStatus($job);
            &smEnd;
    }
    }//if CIM op failed
    else if (0 == #returnStatus)  {
        //completed successfully
        &smCommandCompleted($job);
        &smEnd;
    }
    else if (0x4096 == #returnStatus) {
        //job spawned, need to watch for it to finish
        //while the jobstate is “Running”
        while (4 == $instanceConcreteJob.JobState){<busy wait>}
        if (2 != $job.OperationalStatus) {
            %InArguments[] = { }
            %OutArguments[] = {newArgument(“Job”, $instanceConcreteJob.getObjectPath())}
            #Error = smOpInvokeMethod($job,
               “GetError”
               %InArguments,
               %OutArguments,
               #returncode);
            //Method invocation failed, internal processing error
            if ( (0 != #Error.code) || (0 != #returncode) )   {
            //make an Error instance and associate with job for Operation
               $OperationError = smNewInstance(“CIM_Error”);
               //CIM_ERR_FAILED
               $OperationError.CIMStatusCode = 1;
               //Software Error
               $OperationError.ErrorType = 4;
               //Unknown
               $OperationError.PerceivedSeverity = 0;
               $OperationError.OwningEntity = DMTF:SMCLP;
               $OperationError.MessageID = 0x00000009;
               $OperationError.Message = “An internal software error has occurred.”;
               &smAddError($job, $OperationError);
               &smMakeCommandStatus($job);
               &smEnd;
            }
            else   {
               //make command status
               $joberror = %OutArguments[“Error”];
               &smCommandExecutionFailed($job, {$joberror};
            }//end if have CIM_Error from GetError()
        }//embedded job not OK
        else {
            //the job ran to completion (we assume)
            &smCommandComplete($job);
          &smEnd;
        }
    }//if job spawned
    else if (1 == #returnStatus)  {
        //unsupported
        $OperationError = smNewInstance(“CIM_Error”);
        //CIM_ERR_NOT_SUPPORTED
        $OperationError.CIMStatusCode = 7;
        //Other
        $OperationError.ErrorType = 1;
        //Low
        $OperationError.PerceivedSeverity = 2;
        $OperationError.OwningEntity = DMTF:SMCLP;
        $OperationError.MessageID = 0x00000001;
        $OperationError.Message = “Operation is not supported.”;
        &smAddError($job, $OperationError);
```

```
        &smMakeCommandStatus($job);
        &smEnd;
    }
    else if (5 == #returnStatus) {
        //unsupported
        $OperationError = smNewInstance("CIM_Error");
        //CIM_ERR_INVALID_PARAMETER
        $OperationError.CIMStatusCode = 4;
        //Other
        $OperationError.ErrorType = 1;
        //Low
        $OperationError.PerceivedSeverity = 2;
        $OperationError.OwningEntity = DMTF:SMCLP;
        $OperationError.MessageID = 0x00000004;
        $OperationError.Message = "One or more parameters specified are invalid.";
        &smAddError($job, $OperationError);
        &smMakeCommandStatus($job);
        &smEnd;
    }
    else if (6 == #returnStatus || 4099 == #returnStatus) {
        //busy
        $OperationError = smNewInstance("CIM_Error");
        //CIM_ERR_FAILED
        $OperationError.CIMStatusCode = 1;
        //Other
        $OperationError.ErrorType = 1;
        //Low
        $OperationError.PerceivedSeverity = 2;
        $OperationError.OwningEntity = DMTF:SMCLP;
        $OperationError.MessageID = 0x0000000A;
        $OperationError.Message = "The target is busy and its state cannot be
           changed.";
        &smAddError($job, $OperationError);
        &smMakeCommandStatus($job);
        &smEnd;
    }
    else if (4097 == $returnStatus)   {
        //invalid state transition
        $OperationError = smNewInstance("CIM_Error");
        //CIM_ERR_FAILED
        $OperationError.CIMStatusCode = 1;
        //Other
        $OperationError.ErrorType = 1;
        //Low
        $OperationError.PerceivedSeverity = 2;
        $OperationError.OwningEntity = DMTF:SMCLP;
        $OperationError.MessageID = 0x0000000B;
        $OperationError.Message = "The target cannot transition to the requested state
           from its current state.";
        &smAddError($job, $OperationError);
        &smMakeCommandStatus($job);
    }
    else if (2 == #returnStatus || 4 == #returnStatus || 3 == $returnStatus )  {
        //generic failure
        $OperationError = smNewInstance(“CIM_Error”);
        //CIM_ERR_FAILED
        $OperationError.CIMStatusCode = 1;
        //Other
        $OperationError.ErrorType = 1;
        //Low
        $OperationError.PerceivedSeverity = 2;
        $OperationError.OwningEntity = DMTF:SMCLP;
        $OperationError.MessageID = 0x00000002;
        $OperationError.Message = “Failed. No further information is available.”;
        &smAddError($job, $OperationError);
        &smMakeCommandStatus($job);
    }
    else   {
        //unspecified return code, generic failure
        $OperationError = smNewInstance(“CIM_Error”);
        //CIM_ERR_FAILED
        $OperationError.CIMStatusCode = 1;
        //Other
        $OperationError.ErrorType = 1;
        //Low
        $OperationError.PerceivedSeverity = 2;
        $OperationError.OwningEntity = DMTF:SMCLP;
        $OperationError.MessageID = 0x00000002;
        $OperationError.Message = “Failed. No further information is available.”;
        &smAddError($job, $OperationError);
        &smMakeCommandStatus($job);
        &smEnd;
    }
}//end smRequestStateChange()
```

# 6 Mappings

The following sections detail the mapping of CLP verbs to CIM Operations for each CIM class defined in the *Base Metrics Profile*. Requirements specified here related to support for a CLP verb for a particular class are solely within the context of this profile.

## 6.1 CIM_OperatingSystem

The `cd`, `help`, `version`, and `exit` verbs shall be supported as described in DSP0216.

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

**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 | |

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

### 6.1.1 Ordering of Results

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

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

### 6.1.2 Show

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

#### 6.1.2.1 Show Command Form for a Single Instance – CIM_MetricService Reference

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

##### 6.1.2.1.1 Command Form

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

##### 6.1.2.1.2 CIM Requirements

See CIM_ElementCapabilities in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

###### 6.1.2.1.3 Behavior Requirements

###### 6.1.2.1.3.1 Preconditions

$instance represents the instance of a CIM_MetricService, which is referenced by CIM_ElementCapabilities.

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

###### 6.1.2.1.3.2 Pseudo Code

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

#### 6.1.2.2 Show Command Form for Multiple Instances – CIM_MetricServiceCapabilities Reference

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

##### 6.1.2.2.1 Command Form

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

##### 6.1.2.2.2 CIM Requirements

See CIM_ElementCapabilities in the "CIM Elements" section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.1.2.2.3 Behavior Requirements

###### 6.1.2.2.3.1 Preconditions

$instance represents the instance of a CIM_MetricServiceCapabilities, which is referenced by CIM_ElementCapabilities.

###### 6.1.2.2.3.2 Pseudo Code

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

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

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

##### 6.1.2.3.1 Command Form

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

##### 6.1.2.3.2 CIM Requirements

See CIM_ElementCapabilities in the "CIM Elements" section of the *Base Metrics Profile* for the list of mandatory properties.

#### 6.1.2.3.3 Behavior Requirements

##### 6.1.2.3.3.1 Preconditions

$instanceA represents the instance of a CIM_MetricService which is referenced by CIM_ElementCapabilities.

$instanceB represents the instance of a CIM_MetricServiceCapabilities or CIM_EnabledLogicalElementCapabilities which is referenced by CIM_ElementCapabilities.

##### 6.1.2.3.3.2 Pseudo Code

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

## 6.2 CIM_HostedService

The `cd` and `help` verbs shall be supported as described in DSP0216.

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

**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 | |

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

### 6.2.1 Ordering of Results

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

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

### 6.2.2 Show

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

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

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

##### 6.2.2.1.1 Command Form

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

##### 6.2.2.1.2 CIM Requirements

See CIM_HostedService in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.2.2.1.3 Behavior Requirements

###### 6.2.2.1.3.1 Preconditions

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

###### 6.2.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_HostedService”, $instance.getObjectPath() );
&smEnd;
```

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

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

##### 6.2.2.2.1 Command Form

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

##### 6.2.2.2.2 CIM Requirements

See CIM_HostedService in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.2.2.2.3 Behavior Requirements

###### 6.2.2.2.3.1 Preconditions

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

###### 6.2.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances (“CIM_HostedService”, $instance.getObjectPath() );
&smEnd;
```

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

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

##### 6.2.2.3.1 Command Form

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

##### 6.2.2.3.2 CIM Requirements

See CIM_HostedService in the "CIM Elements" section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.2.2.3.3 Behavior Requirements

###### 6.2.2.3.3.1 Preconditions

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

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

###### 6.2.2.3.3.2 Pseudo Code

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

## 6.3 CIM_MetricInstance

The `cd` and `help` verbs shall be supported as described in DSP0216.

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

**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 | |

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

### 6.3.1 Ordering of Results

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

- Results for CIM_MetricInstance are unordered; therefore, no algorithm is defined.

### 6.3.2 Show

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

#### 6.3.2.1 Show Command Form for Multiple Instances – CIM_BaseMetricDefinition or CIM_AggregationMetricDefinition Reference

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

##### 6.3.2.1.1 Command Form

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

##### 6.3.2.1.2 CIM Requirements

See CIM_MetricInstance in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.3.2.1.3 Behavior Requirements

###### 6.3.2.1.3.1 Preconditions

$instance represents the instance of CIM_BaseMetricDefinition or CIM_AggregationMetricDefinition, which is referenced by CIM_MetricInstance.

###### 6.3.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_MetricInstance”, $instance.getObjectPath() );
&smEnd;
```

#### 6.3.2.2 Show Command Form for a Single Instance – CIM_BaseMetricValue or CIM_AggregationMetricValue Reference

This command form is when the `show` verb applies to a single instance. The command form corresponds to the `show` verb issued against instances of CIM_MetricInstance where only one reference is specified and the reference is to an instance of CIM_BaseMetricValue or CIM_AggregationMetricValue.

##### 6.3.2.2.1 Command Form

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

##### 6.3.2.2.2 CIM Requirements

See CIM_MetricInstance in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

###### 6.3.2.2.3 Behavior Requirements

###### 6.3.2.2.3.1 Preconditions

$instance represents the instance of CIM_BaseMetricValue or CIM_AggregationMetricValue, which is referenced by CIM_MetricInstance.

###### 6.3.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances (“CIM_MetricInstance”, $instance.getObjectPath() );
&smEnd;
```

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

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

##### 6.3.2.3.1 Command Form

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

##### 6.3.2.3.2 CIM Requirements

See CIM_MetricInstance in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.3.2.3.3 Behavior Requirements

###### 6.3.2.3.3.1 Preconditions

$instanceA represents the referenced instance of CIM_BaseMetricDefinition or CIM_AggregationMetricDefinition through the CIM_MetricInstance association.

$instanceB represents the other instance of CIM_BaseMetricValue or CIM_AggregationMetricValue which is referenced by CIM_MetricInstance.

###### 6.3.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance (“CIM_MetricInstance”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath() );
&smEnd;
```

## 6.4 CIM_ServiceAffectsElement

The `cd` and `help` verbs shall be supported as described in DSP0216.

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

**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 | |

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

### 6.4.1 Ordering of Results

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

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

### 6.4.2 Show

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

#### 6.4.2.1 Show Command Form for a Single Instance – CIM_BaseMetricDefinition or CIM_AggregationMetricDefinition Reference

This command form is used when the `show` verb applies to a single instance. This command form corresponds to a `show` command issued against an instance of CIM_ServiceAffectsElement where only one reference is specified and the reference is to an instance of CIM_BaseMetricDefinition or CIM_AggregationMetricDefinition.

##### 6.4.2.1.1 Command Form

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

##### 6.4.2.1.2 CIM Requirements

See CIM_ServiceAffectsElement in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.4.2.1.3 Behavior Requirements

###### 6.4.2.1.3.1 Preconditions

$instance represents the instance of CIM_BaseMetricDefinition or CIM_AggregationMetricDefinition, which is referenced by CIM_ServiceAffectsElement.

###### 6.4.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ServiceAffectsElement”,
   $instance.getObjectPath() );
&smEnd;
```

### 6.4.2.2 Show Command Form for Multiple Instances – CIM_MetricService Reference

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

#### 6.4.2.2.1 Command Form

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

#### 6.4.2.2.2 CIM Requirements

See CIM_ServiceAffectsElement in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

#### 6.4.2.2.3 Behavior Requirements

##### 6.4.2.2.3.1 Preconditions

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

##### 6.4.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ServiceAffectsElement”,
   $instance.getObjectPath() );
&smEnd;
```

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

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

#### 6.4.2.3.1 Command Form

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

#### 6.4.2.3.2 CIM Requirements

See CIM_ServiceAffectsElement in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

#### 6.4.2.3.3 Behavior Requirements

##### 6.4.2.3.3.1 Preconditions

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

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

#### 6.4.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_ServiceAffectsElement”, $instanceA.getObjectPath(),
    $instanceB.getObjectPath() );
&smEnd;
```

## 6.5 CIM_MetricDefForME

The `cd` and `help` verbs shall be supported as described in DSP0216.

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

**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 | |

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

### 6.5.1 Ordering of Results

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

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

### 6.5.2 Show

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

#### 6.5.2.1 Show Command Form for Multiple Instances – CIM_BaseMetricDefinition or CIM_AggregationMetricDefinition Reference

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

##### 6.5.2.1.1 Command Form

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

##### 6.5.2.1.2 CIM Requirements

See CIM_MetricDefForME in the “CIM Elements” section of the Base Metrics Profile for the list of mandatory properties.

##### 6.5.2.1.3 Behavior Requirements

###### 6.5.2.1.3.1 Preconditions

$instance represents the instance of CIM_BaseMetricDefinition or CIM_AggregationMetricDefinition, which is referenced by CIM_MetricDefForME.

###### 6.5.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_MetricDefForME”, $instance.getObjectPath() );
&smEnd;
```

#### 6.5.2.2 Show Command Form for Multiple Instances – CIM_ManagedElement Reference

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

##### 6.5.2.2.1 Command Form

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

##### 6.5.2.2.2 CIM Requirements

See CIM_MetricDefForME in the “CIM Elements” section of the Base Metrics Profile for the list of mandatory properties.

##### 6.5.2.2.3 Behavior Requirements

###### 6.5.2.2.3.1 Preconditions

$instance represents the instance of CIM_ManagedElement, which is referenced by CIM_MetricDefForME.

###### 6.5.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_MetricDefForME”, $instance.getObjectPath() );
&smEnd;
```

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

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

##### 6.5.2.3.1 Command Form

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

#### 6.5.2.3.2 CIM Requirements

See CIM_MetricDefForME in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

#### 6.5.2.3.3 Behavior Requirements

##### 6.5.2.3.3.1 Preconditions

$instanceA represents the instance of CIM_BaseMetricDefinition or CIM_AggregationMetricDefinition, which is referenced by CIM_MetricDefForME.

$instanceB represents the instance CIM_ManagedElement which is referenced by CIM_MetricDefForME.

##### 6.5.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_MetricDefForME”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath() );
&smEnd;
```

## 6.6 CIM_MetricsForME

The `cd` and `help` verbs shall be supported as described in DSP0216.

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

**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 | |

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

### 6.6.1 Ordering of Results

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

- Results for CIM_MetricForME are unordered; therefore, no algorithm is defined.

### 6.6.2 Show

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

#### 6.6.2.1 Show Command Form for Multiple Instances – CIM_ManagedElement Reference

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

##### 6.6.2.1.1 Command Form

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

##### 6.6.2.1.2 CIM Requirements

See CIM_MetricForME in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.6.2.1.3 Behavior Requirements

###### 6.6.2.1.3.1 Preconditions

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

###### 6.6.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances (“CIM_MetricForME”, $instance.getObjectPath() );
&smEnd;
```

#### 6.6.2.2 Show Command Form for Multiple Instances – CIM_BaseMetricValue or CIM_AggregationMetricValue Reference

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

##### 6.6.2.2.1 Command Form

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

##### 6.6.2.2.2 CIM Requirements

See CIM_MetricForME in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.6.2.2.3 Behavior Requirements

###### 6.6.2.2.3.1 Preconditions

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

###### 6.6.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances (“CIM_MetricForME”, $instance.getObjectPath() );
&smEnd;
```

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

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

###### 6.6.2.3.1 Command Form

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

###### 6.6.2.3.2 CIM Requirements

See CIM_MetricForME in the "CIM Elements" section of the *Base Metrics Profile* for the list of mandatory properties.

###### 6.6.2.3.3 Behavior Requirements

###### 6.6.2.3.3.1 Preconditions

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

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

###### 6.6.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ("CIM_MetricForME", $instanceA.getObjectPath(),
    $instanceB.getObjectPath() );
&smEnd;
```

## 6.7 CIM_ConcreteDependency

The cd and help verbs shall be supported as described in DSP0216.

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

**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 | |

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

### 6.7.1 Ordering of Results

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

- Results for CIM_ConcreteDependency are unordered; therefore, no algorithm is defined.

### 6.7.2 Show

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

#### 6.7.2.1 Show Command Form for a Single Instance – CIM_AggregationMetricValue Reference

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

##### 6.7.2.1.1 Command Form

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

##### 6.7.2.1.2 CIM Requirements

See CIM_ConcreteDependency in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.7.2.1.3 Behavior Requirements

###### 6.7.2.1.3.1 Preconditions

$instance represents the instance of CIM_AggregationMetricValue, which is referenced by CIM_ConcreteDependency.

###### 6.7.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ConcreteDependency”, $instance.getObjectPath() );
&smEnd;
```

#### 6.7.2.2 Show Command Form for a Single Instance – CIM_AggregationMetricDefinition Reference

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

##### 6.7.2.2.1 Command Form

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

##### 6.7.2.2.2 CIM Requirements

See CIM_ConcreteDependency in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

#### 6.7.2.2.3 Behavior Requirements

##### 6.7.2.2.3.1 Preconditions

$instance represents the instance of CIM_AggregationMetricDefinition, which is referenced by CIM_ConcreteDependency.

##### 6.7.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ConcreteDependency”, $instance.getObjectPath() );
&smEnd;
```

### 6.7.2.3 Show Command Form for Multiple Instances – CIM_BaseMetricValue Reference

This command form is when the show verb applies to multiple instances. The command form corresponds to the show verb issued against instances of CIM_ConcreteDependency where only one reference is specified and the reference is to an instance of CIM_BaseMetricValue.

#### 6.7.2.3.1 Command Form

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

#### 6.7.2.3.2 CIM Requirements

See CIM_ConcreteDependency in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

#### 6.7.2.3.3 Behavior Requirements

##### 6.7.2.3.3.1 Preconditions

$instance represents the instance of CIM_BaseMetricValue, which is referenced by CIM_ConcreteDependency.

##### 6.7.2.3.3.2 Pseudo Code

```
&smShowAssociationInstances (“CIM_ConcreteDependency”, $instance.getObjectPath() );
&smEnd;
```

### 6.7.2.4 Show Command Form for Multiple Instances – CIM_BaseMetricDefinition Reference

This command form is when the show verb applies to multiple instances. The command form corresponds to the show verb issued against instances of CIM_ConcreteDependency where only one reference is specified and the reference is to an instance of CIM_BaseMetricDefinition.

#### 6.7.2.4.1 Command Form

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

#### 6.7.2.4.2 CIM Requirements

See CIM_ConcreteDependency in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

#### 6.7.2.4.3 Behavior Requirements

##### 6.7.2.4.3.1 Preconditions

$instance represents the instance of CIM_BaseMetricDefinition, which is referenced by CIM_ConcreteDependency.

##### 6.7.2.4.3.2 Pseudo Code

```
&smShowAssociationInstances (“CIM_ConcreteDependency”, $instance.getObjectPath() );
&smEnd;
```

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

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

#### 6.7.2.5.1 Command Form

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

#### 6.7.2.5.2 CIM Requirements

See CIM_ConcreteDependency in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

#### 6.7.2.5.3 Behavior Requirements

##### 6.7.2.5.3.1 Preconditions

$instanceA represents the referenced instance of CIM_AggregationMetricValue or CIM_AggregationMetricDefinition through CIM_ConcreteDependency association.

$instanceB represents the other instance of CIM_BaseMetricValue or CIM_BaseMetricDefinition which is referenced by CIM_ConcreteDependency.

##### 6.7.2.5.3.2 Pseudo Code

```
&smShowAssociationInstance (“CIM_ConcreteDependency”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath() );
&smEnd;
```

## 6.8 CIM_BaseMetricDefinition

The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in DSP0216.

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

**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. |

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

### 6.8.1 Ordering of Results

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

- Results for CIM_BaseMetricDefinition are unordered; therefore, no algorithm is defined.

### 6.8.2 Reset

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

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

##### 6.8.2.1.1 Command Form

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

##### 6.8.2.1.2 CIM Requirements

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

##### 6.8.2.1.3 Behavior Requirements

###### 6.8.2.1.3.1 Preconditions

$instance represents the targeted instance of CIM_BaseMetricDefinition.

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

###### 6.8.2.1.3.2 Pseudo Code

```
lControlMetrics ( $instance.getObjectPath(), “Reset” ) ();
&smEnd;
```

### 6.8.3 Set

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

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

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

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

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Fan Profile*.

##### 6.8.3.1.1 Command Form

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

##### 6.8.3.1.2 CIM Requirements

See CIM_BaseMetricDefinition in the “CIM Elements” section of the *Fan Profile* for the list of mandatory properties.

##### 6.8.3.1.3 Behavior Requirements

###### 6.8.3.1.3.1 Preconditions

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

###### 6.8.3.1.3.2 Pseudo Code

```
#propertyNames[] = {<propertyname>};
#propertyValues[] = {<propertyvalue>};
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

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

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

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Fan Profile*.

##### 6.8.3.2.1 Command Form

```
set <CIM_BaseMetricDefinition single instance>
   <propertyname1>=<propertyvalue1><propertynamen>=<propertyvaluen>
```

##### 6.8.3.2.2 CIM Requirements

See CIM_BaseMetricDefinition in the “CIM Elements” section of the *Fan Profile* for the list of mandatory properties.

##### 6.8.3.2.3 Behavior Requirements

###### 6.8.3.2.3.1 Preconditions

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

###### 6.8.3.2.3.2 Pseudo Code

```
#propertyNames[] = {<propertyname>};
for #i < n
{
   #propertyNames[#i] = <propertname#i>
   #propertyValues[#i] = <propertyvalue#i>
}
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

### 6.8.4 Show

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

#### 6.8.4.1 Show Command Form for Multiple Instances Target

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

##### 6.8.4.1.1 Command Form

**`show <CIM_BaseMetricDefinition` *`multiple instances`*`>`**

##### 6.8.4.1.2 CIM Requirements

See CIM_BaseMetricDefinition in the “CIM Elements” section of the *Fan Profile* for the list of mandatory properties.

##### 6.8.4.1.3 Behavior Requirements

###### 6.8.4.1.3.1 Preconditions

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

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

###### 6.8.4.1.3.2 Pseudo Code

```
#Error=smOpAsociators(
   $containerinstance->,
   “CIM_ServiceAffectsElement”,
   NULL,
   NULL,
   NULL,
   $definitionInstancePaths[])
if (0 != #Error.code)
```

```
   {
   &smProcessOpError (#Error);
   //includes &smEnd;
   }
else
   {
for #i < $definitionInstancePaths.length
 {
// the class definition for $instance includes two referenced properties,
// MetricCollectionEnabled and RecordedSince.
#Error=smOpReferences(
   $definitionInstancePaths->[i],
   “CIM_MetricDefForME”,
   NULL,
   NULL,
   {“MetricCollectionEnabled”,“RecordedSince”},
   $MDFMEInstancePaths[])
if (0 != #Error.code)
   {
   &smProcessOpError (#Error);
   //includes &smEnd;
   }
else
   {
   #propertynamelist[] = null;
   if ( false == #all) {
   #propertynamelist[] = <array of mandatory non-key property names (see CIM
   Requirements)>;
   }
#additionalpropertylist[]={“MetricCollectionEnabled”,“RecordedSince”};
$MDFMEInstance = $MDFMEInstancePaths[1];
$instance.MetricCollectionEnabled =$APMSinstance.MetricCollectionEnabled;
$instance.RecordedSince =$APMSinstance.RecordedSince;
&smShowInstancePseudoProperties(
   $instance,
   #propertynamelist[],
   #additionalpropertylist[] );
}
   i++;
}
&smEnd;
```

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

This command form is used to show a single instance of CIM_BaseMetricDefinition.

##### 6.8.4.2.1 Command Form

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

##### 6.8.4.2.2 CIM Requirements

See CIM_BaseMetricDefinition in the "CIM Elements" section of the *Fan Profile* for the list of mandatory properties.

##### 6.8.4.2.3 Behavior Requirements

###### 6.8.4.2.3.1 Preconditions

$instance represents the targeted instance of CIM_BaseMetricDefinition.

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

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

###### 6.8.4.2.3.2 Pseudo Code

```
// the class definition for $instance includes two referenced properties,
// MetricCollectionEnabled and RecordedSince.
#Error=smOpReferences (
   $instance->,
   "CIM_MetricDefForME",
   NULL,
   NULL,
   {"MetricCollectionEnabled","RecordedSince"},
   $MDFMEInstancePaths[] )
if (0 != #Error.code)
   {
   &smProcessOpError (#Error);
   //includes &smEnd;
   }
else
   {
   #propertynamelist[] = null;
   if ( false == #all)
      {
      #propertynamelist[] = <array of mandatory non-key property names (see CIM
         Requirements)>;
      }
   #additionalpropertylist[]={"MetricCollectionEnabled","RecordedSince"};
   $MDFMEInstance=$MDFMEInstancePaths[1];
   $instance.MetricCollectionEnabled=$APMSinstance.MetricCollectionEnabled;
   $instance.RecordedSince=$APMSinstance.RecordedSince;
   &smShowInstancePseudoProperties (
      $instance,
      #propertynamelist[],
      #additionalpropertylist[] );
   }
&smEnd;
```

### 6.8.5 Start

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

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

##### 6.8.5.1.1 Command Form

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

##### 6.8.5.1.2 CIM Requirements

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

##### 6.8.5.1.3 Behavior Requirements

###### 6.8.5.1.3.1 Preconditions

$instance represents the targeted instance of CIM_BaseMetricDefinition.

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

###### 6.8.5.1.3.2 Pseudo Code

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

### 6.8.6 Stop

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

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

##### 6.8.6.1.1 Command Form

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

##### 6.8.6.1.2 CIM Requirements

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

##### 6.8.6.1.3 Behavior Requirements

###### 6.8.6.1.3.1 Preconditions

$instance represents the targeted instance of CIM_BaseMetricDefinition.

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

#### 6.8.6.1.3.2 Pseudo Code

```
lControlMetrics ( $instance.getObjectPath(), “Disable” ) ();
&smEnd;
```

## 6.9 CIM_BaseMetricValue

The `cd` and `help` verbs shall be supported as described in DSP0216.

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

**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 | |

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

### 6.9.1 Ordering of Results

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

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

### 6.9.2 Set

#### 6.9.2.1 General Usage of Set for a Single Property

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

The requirements for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Base Metrics Profile*.

##### 6.9.2.1.1 Command Form

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

#### 6.9.2.1.2 CIM Requirements

See CIM_BaseMetricValue in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

#### 6.9.2.1.3 Behavior Requirements

##### 6.9.2.1.3.1 Preconditions

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

##### 6.9.2.1.3.2 Pseudo Code

```
#propertyNames[] = <propertname>
#propertyValues[] = <propertyvalue>
&smSetInstance ( $instance, #propertyNames, #propertyValues );
&smEnd;
```

### 6.9.2.2 General Usage of Set for Multiple Properties

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

The requirements for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Base Metrics Profile*.

#### 6.9.2.2.1 Command Form

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

#### 6.9.2.2.2 CIM Requirements

See CIM_BaseMetricValue in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

#### 6.9.2.2.3 Behavior Requirements

##### 6.9.2.2.3.1 Preconditions

$instance represents the instance of CIM_BaseMetricValue.

##### 6.9.2.2.3.2 Pseudo Code

```
for #i < n
   {
   #propertyNames[#i] = <propertname#i>
   #propertyValues[#i] = <propertyvalue#i>
   }
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

## 6.9.3 Show

The show verb is used to display information about instances of CIM_BaseMetricValue. Implementations shall support the use of the show verb with CIM_BaseMetricValue.

#### 6.9.3.1 Show a Single Instance

This command form is used to display the information about a single instance of CIM_BaseMetricValue.

##### 6.9.3.1.1 Command Form

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

##### 6.9.3.1.2 CIM Requirements

See CIM_BaseMetricValue in the "CIM Elements" section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.9.3.1.3 Behavior Requirements

###### 6.9.3.1.3.1 Preconditions

$instance represents the instance of CIM_BaseMetricValue.

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

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

###### 6.9.3.1.3.2 Pseudo Code

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

#### 6.9.3.2 Show Multiple Instances

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

##### 6.9.3.2.1 Command Form

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

##### 6.9.3.2.2 CIM Requirements

See CIM_BaseMetricValue in the "CIM Elements" section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.9.3.2.3 Behavior Requirements

###### 6.9.3.2.3.1 Preconditions

$containerInstance represents the instance of CIM_BaseMetricDefinition to which the instance of CIM_BaseMetricValue being displayed is scoped. The CIM_BaseMetricDefinition is associated to targeted instances of CIM_BaseMetricValue via a CIM_MetricInstance association.

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

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

#### 6.9.3.2.3.2 Pseudo Code

```
if (false != #all) { #propertylist[] = NULL; }
&smShowInstances ( “CIM_BaseMetricValue”, “CIM_MetricInstance”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

## 6.10 CIM_AggregationMetricDefinition

The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in DSP0216.

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

**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. |

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

### 6.10.1 Ordering of Results

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

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

### 6.10.2 Reset

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

#### 6.10.2.1 General Usage of Reset for a Single Property

##### 6.10.2.1.1 Command Form

```
reset <CIM_AggregationMetricDefinition single instance>
```

#### 6.10.2.1.2 CIM Requirements

```
uint16 CIM_MetricDefForME.MetricCollectionEnabled;
uint32 CIM_MetricService.ControlMetrics(
   [IN] CIM_ManagedElement REF Subject,
   [IN] CIM_AggregationMetricDefinition REF Definition,
   [IN] uint16 MetricCollectionEnabled );
```

#### 6.10.2.1.3 Behavior Requirements

##### 6.10.2.1.3.1 Preconditions

$instance represents the targeted instance of CIM_AggregationMetricDefinition.

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

##### 6.10.2.1.3.2 Pseudo Code

```
lControlMetrics ( $instance.getObjectPath(), “Reset” ) ();
&smEnd;
```

## 6.10.3 Set

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

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

### 6.10.3.1 General Usage of Set for a Single Property

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

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Fan Profile*.

#### 6.10.3.1.1 Command Form

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

#### 6.10.3.1.2 CIM Requirements

See CIM_AggregationMetricDefinition in the “CIM Elements” section of the *Fan Profile* for the list of mandatory properties.

#### 6.10.3.1.3 Behavior Requirements

##### 6.10.3.1.3.1 Preconditions

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

##### 6.10.3.1.3.2 Pseudo Code

```
#propertyNames[] = {<propertyname>};
#propertyValues[] = {<propertyvalue>};
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

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

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

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Fan Profile*.

##### 6.10.3.2.1 Command Form

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

##### 6.10.3.2.2 CIM Requirements

See CIM_AggregationMetricDefinition in the “CIM Elements” section of the *Fan Profile* for the list of mandatory properties.

##### 6.10.3.2.3 Behavior Requirements

###### 6.10.3.2.3.1 Preconditions

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

###### 6.10.3.2.3.2 Pseudo Code

```
#propertyNames[] = {<propertyname>};
for #i < n
   {
   #propertyNames[#i] = <propertname#i>
   #propertyValues[#i] = <propertyvalue#i>
   }
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

### 6.10.4 Show

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

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

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

##### 6.10.4.1.1 Command Form

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

##### 6.10.4.1.2 CIM Requirements

See CIM_AggregationMetricDefinition in the “CIM Elements” section of the *Fan Profile* for the list of mandatory properties.

#### 6.10.4.1.3 Behavior Requirements

##### 6.10.4.1.3.1 Preconditions

$containerInstance represents the instance of CIM_MetricService which represents the container service and is associated to the targeted instances of CIM_AggregationMetricDefinition through the CIM_ServiceAffectsElement association.

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

##### 6.10.4.1.3.2 Pseudo Code

```
#Error=smOpAsociators (
   $containerinstance->,
   “CIM_ServiceAffectsElement”,
   NULL,
   NULL,
   NULL,
   $definitionInstancePaths[] )
if (0 != #Error.code)
   {
   &smProcessOpError (#Error);
   //includes &smEnd;
   }
else
   {
      for #i < $definitionInstancePaths.length
      {
      // the class definition for $instance includes two referenced properties,
      // MetricCollectionEnabled and RecordedSince.
      #Error=smOpReferences(
         $definitionInstancePaths->[i],
         “CIM_MetricDefForME”,
         NULL,
         NULL,
         {“MetricCollectionEnabled”,“RecordedSince”},
         $MDFMEInstancePaths[])
      if (0 != #Error.code)
      {
         &smProcessOpError (#Error);
         //includes &smEnd;
      }
      else
      {
         #propertynamelist[] = null;
         if ( false == #all) {
            #propertynamelist[] = <array of mandatory non-key property names (see CIM
            Requirements)>;
         }
      #additionalpropertylist[]={“MetricCollectionEnabled”,“RecordedSince”};
      $MDFMEInstance = $MDFMEInstancePaths[1];
      $instance.MetricCollectionEnabled
      =$APMSinstance.MetricCollectionEnabled;
      $instance.RecordedSince =$APMSinstance.RecordedSince;
      &smShowInstancePseudoProperties(
         $instance,
         #propertynamelist[],
         #additionalpropertylist[] );
   }
   i++;
   }
&smEnd;
```

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

This command form is used to show a single instance of CIM_AggregationMetricDefinition.

#### 6.10.4.2.1 Command Form

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

#### 6.10.4.2.2 CIM Requirements

See CIM_AggregationMetricDefinition in the “CIM Elements” section of the *Fan Profile* for the list of mandatory properties.

#### 6.10.4.2.3 Behavior Requirements

##### 6.10.4.2.3.1 Preconditions

$instance represents the targeted instance of CIM_AggregationMetricDefinition.

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

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

##### 6.10.4.2.3.2 Pseudo Code

```
// the class definition for $instance includes two referenced properties,
// MetricCollectionEnabled and RecordedSince.
#Error=smOpReferences (
   $instance->,
   “CIM_MetricDefForME”,
   NULL,
   NULL,
   {“MetricCollectionEnabled”,“RecordedSince”},
   $MDFMEInstancePaths[] )
if (0 != #Error.code)
   {
   &smProcessOpError (#Error);
   //includes &smEnd;
   }
else
   {
      #propertynamelist[] = null;
      if ( false == #all)
      {
      #propertynamelist[] = <array of mandatory non-key property names (see CIM
         Requirements)>;
      }
   #additionalpropertylist[]={“MetricCollectionEnabled”,“RecordedSince”};
   $MDFMEInstance=$MDFMEInstancePaths[1];
   $instance.MetricCollectionEnabled=$APMSinstance.MetricCollectionEnabled;
   $instance.RecordedSince=$APMSinstance.RecordedSince;
   &smShowInstancePseudoProperties(
      $instance,
      #propertynamelist[],
      #additionalpropertylist[]);
   }
&smEnd;
```

### 6.10.5 Start

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

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

##### 6.10.5.1.1 Command Form

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

##### 6.10.5.1.2 CIM Requirements

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

##### 6.10.5.1.3 Behavior Requirements

###### 6.10.5.1.3.1 Preconditions

$instance represents the targeted instance of CIM_AggregationMetricDefinition.

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

###### 6.10.5.1.3.2 Pseudo Code

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

### 6.10.6 Stop

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

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

##### 6.10.6.1.1 Command Form

```
stop <CIM_AggregationMetricDefinition single instance>
```

##### 6.10.6.1.2 CIM Requirements

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

##### 6.10.6.1.3 Behavior Requirements

###### 6.10.6.1.3.1 Preconditions

$instance represents the targeted instance of CIM_AggregationMetricDefinition.

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

###### 6.10.6.1.3.2 Pseudo Code

```
lControlMetrics ( $instance.getObjectPath(), “Disable” ) ();
&smEnd;
```

## 6.11 CIM_AggregationMetricValue

The cd and help verbs shall be supported as described in DSP0216.

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

**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 | |

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

### 6.11.1 Ordering of Results

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

- Results for CIM_AggregationMetricValue are unordered; therefore, no algorithm is defined.

### 6.11.2 Set

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

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

The requirements for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Base Metrics Profile*.

##### 6.11.2.1.1 Command Form

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

##### 6.11.2.1.2 CIM Requirements

See CIM_AggregationMetricValue in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.11.2.1.3 Behavior Requirements

###### 6.11.2.1.3.1 Preconditions

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

###### 6.11.2.1.3.2 Pseudo Code

```
#propertyNames[] = <propertname>
#propertyValues[] = <propertyvalue>
&smSetInstance ( $instance, #propertyNames, #propertyValues );
&smEnd;
```

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

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

The requirements for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Base Metrics Profile*.

##### 6.11.2.2.1 Command Form

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

##### 6.11.2.2.2 CIM Requirements

See CIM_AggregationMetricValue in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.11.2.2.3 Behavior Requirements

###### 6.11.2.2.3.1 Preconditions

$instance represents the instance of CIM_AggregationMetricValue.

###### 6.11.2.2.3.2 Pseudo Code

```
for #i < n
   {
   #propertyNames[#i] = <propertname#i>
   #propertyValues[#i] = <propertyvalue#i>
   }
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

### 6.11.3 Show

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

#### 6.11.3.1 Show a Single Instance

This command form is used to display the information about a single instance of CIM_AggregationMetricValue.

##### 6.11.3.1.1 Command Form

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

##### 6.11.3.1.2 CIM Requirements

See CIM_AggregationMetricValue in the "CIM Elements" section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.11.3.1.3 Behavior Requirements

###### 6.11.3.1.3.1 Preconditions

$instance represents the instance of CIM_AggregationMetricValue.

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

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

###### 6.11.3.1.3.2 Pseudo Code

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

#### 6.11.3.2 Show Multiple Instances

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

#### 6.11.3.2.1 Command Form

```
show <CIM_AggregationMetricValue multiple instances>
```

#### 6.11.3.2.2 CIM Requirements

See CIM_AggregationMetricValue in the "CIM Elements" section of the *Base Metrics Profile* for the list of mandatory properties.

#### 6.11.3.2.3 Behavior Requirements

##### 6.11.3.2.3.1 Preconditions

$containerInstance represents the instance of CIM_AggregationMetricDefinition to which the instance of CIM_AggregationMetricValue being displayed is scoped. The CIM_AggregationMetricDefinition is associated to targeted instances of CIM_AggregationMetricValue via a CIM_MetricInstance association.

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

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

##### 6.11.3.2.3.2 Pseudo Code

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

## 6.12 CIM_MetricServiceCapabilities

The `cd` and `help` verbs shall be supported as described in DSP0216.

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

**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 | |

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

#### 6.12.1 Ordering of Results

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

- Results for CIM_MetricServiceCapabilities are unordered; therefore, no algorithm is defined.

#### 6.12.2 Show

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

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

##### 6.12.2.1 Show a Single Instance

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

###### 6.12.2.1.1 Command Form

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

###### 6.12.2.1.2 CIM Requirements

See CIM_MetricServiceCapabilities in the "CIM Elements" section of the *Base Metrics Profile* for the list of mandatory properties.

###### 6.12.2.1.3 Behavior Requirements

###### 6.12.2.1.3.1 Preconditions

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

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

###### 6.12.2.1.3.2 Pseudo Code

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

##### 6.12.2.2 Show Multiple Instances

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

###### 6.12.2.2.1 Command Form

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

#### 6.12.2.2.2 CIM Requirements

See CIM_MetricServiceCapabilities in the "CIM Elements" section of the *Base Metrics Profile* for the list of mandatory properties.

#### 6.12.2.2.3 Behavior Requirements

##### 6.12.2.2.3.1 Preconditions

$containerInstance represents the instance of CIM_ConcreteCollection with ElementName property that contains "Capabilities" and is associated to the targeted instances of CIM_MetricServiceCapabilities through the CIM_MemberOfCollection association.

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

##### 6.12.2.2.3.2 Pseudo Code

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

## 6.13 CIM_MetricService

The cd and help verbs shall be supported as described in DSP0216.

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

**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 | |

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

### 6.13.1 Ordering of Results

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

- Results for CIM_MetricService are unordered; therefore, no algorithm is defined.

### 6.13.2 Set

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

Implementations may support the use of the `set` verb with CIM_MetricService.

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

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

The requirements for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Base Metrics Profile*.

##### 6.13.2.1.1 Command Form

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

##### 6.13.2.1.2 CIM Requirements

See CIM_MetricService in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.13.2.1.3 Behavior Requirements

###### 6.13.2.1.3.1 Preconditions

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

###### 6.13.2.1.3.2 Pseudo Code

```
#propertyNames[] = <propertname>
#propertyValues[] = <propertyvalue>
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

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

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

The requirements for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *Base Metrics Profile*.

##### 6.13.2.2.1 Command Form

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

##### 6.13.2.2.2 CIM Requirements

See CIM_MetricService in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.13.2.2.3 Behavior Requirements

###### 6.13.2.2.3.1 Preconditions

$instance represents the instance of CIM_MetricService.

###### 6.13.2.2.3.2 Pseudo Code

```
for #i < n
   {
   #propertyNames[#i] = <propertname#i>
   #propertyValues[#i] = <propertyvalue#i>
   }
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

### 6.13.3 Show

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

#### 6.13.3.1 Show Command Form for a Single Instance

This command form is used to show a single instance of CIM_MetricService.

##### 6.13.3.1.1 Command Form

**`show <CIM_MetricService`** ***`single instance`*****`>`**

##### 6.13.3.1.2 CIM Requirements

See CIM_MetricService in the “CIM Elements” section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.13.3.1.3 Behavior Requirements

###### 6.13.3.1.3.1 Preconditions

$instance represents the instance of CIM_MetricService.

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

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

###### 6.13.3.1.3.2 Pseudo Code

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

#### 6.13.3.2 Show Command Form for Multiple Instances

This command form is used to show multiple instances of CIM_MetricService.

##### 6.13.3.2.1 Command Form

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

##### 6.13.3.2.2 CIM Requirements

See CIM_MetricService in the "CIM Elements" section of the *Base Metrics Profile* for the list of mandatory properties.

##### 6.13.3.2.3 Behavior Requirements

###### 6.13.3.2.3.1 Preconditions

$containerInstance contains the instance of CIM_ComputerSystem that is associated to the targeted instances of CIM_MetricService through the CIM_HostedService association.

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

###### 6.13.3.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all )
   {
   #propertylist[] = {<array of mandatory non-key property names (see CIM
   Requirements)>}
   }
&smShowInstances ( “CIM_MetricService”, “CIM_HostedService”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

## ANNEX A
(informative)

## Change Log

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