Document Number: DSP0837

Date: 2009-06-04

Version: 1.0.0

USB Redirection Profile SM CLP Mapping Specification

Document Type: Specification

Document Status: DMTF Standard

Document Language: E

CONTENTS

Tables

78

Foreword

The *USB Redirection Profile SM CLP Mapping Specification* (DSP0837) 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

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

- Joel Clark – Intel
- Aaron Merkin – IBM
- Jeff Hilland – HP
- Khachatur Papanyan – Dell

Introduction

This document defines the SM CLP mapping for CIM elements described in the *USB Redirection 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 *USB Redirection Profile* using CIM operations.

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

USB Redirection Profile SM CLP 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 *USB Redirection 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 DSP1077, *USB Redirection Profile 1.0*,
http://www.dmtf.org/standards/published_documents/DSP1077_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

2.2 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 selection Tag

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 *SM CLP-to-CIM Common Mapping Specification 1.0* (DSP0216).

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

This mapping does not define any recipes for local reuse.

6 Mappings

The following sections detail the mapping of CLP verbs to CIM Operations for each CIM class defined in the *USB Redirection 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_ElementCapabilities

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 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.

204 **Table 1 – Command Verb Requirements for CIM_ElementCapabilities**

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

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

207 ### 6.1.1 Ordering of Results

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

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

211 ### 6.1.2 Show

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

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

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

221 ##### 6.1.2.1.1 Command Form

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

223 ##### 6.1.2.1.2 CIM Requirements

224 See CIM_ElementCapabilities in the “CIM Elements” section of the *USB Redirection Profile* for the list of
225 mandatory properties.

226 ##### 6.1.2.1.3 Behavior Requirements

227 ###### 6.1.2.1.3.1 Preconditions

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

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

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_USBRedirectionCapabilities Reference

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

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 *USB Redirection 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_USBRedirectionServiceCapabilities, 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 Command Form for Multiple Instances – CIM_EnabledLogicalElementCapabilities Reference

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

6.1.2.3.1 Command Form

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

6.1.2.3.2 CIM Requirements

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

6.1.2.3.3 Behavior Requirements

6.1.2.3.3.1 Preconditions

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

6.1.2.3.3.2 Pseudo Code

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

6.1.2.4 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 the `show` command issued against instances of CIM_ElementCapabilities where both references are specified; therefore, the desired instance is unambiguously identified.

6.1.2.4.1 Command Form

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

6.1.2.4.2 CIM Requirements

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

6.1.2.4.3 Behavior Requirements

6.1.2.4.3.1 Preconditions

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

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

6.1.2.4.3.2 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 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	

Command Verb	Requirement	Comments
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 the `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 *USB Redirection 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_USBRedirectionService 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_USBRedirectionService.

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 *USB Redirection 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_USBRedirectionService, 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 the `show` command issued against CIM_HostedService where both references are specified; 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 *USB Redirection 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 CIM_HostedService association.

`$instanceB` represents the other instance of CIM_USBRedirectionService 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_HostedAccessPoint

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 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

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

364 **Table 3 – Command Verb Requirements for CIM_HostedAccessPoint**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.3.2.
start	Not supported	
stop	Not supported	

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

367 ### 6.3.1 Ordering of Results

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

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

371 ### 6.3.2 Show

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

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

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

379 ##### 6.3.2.1.1 Command Form

380 ```
show <CIM_HostedAccessPoint multiple instances>
```

381 ##### 6.3.2.1.2 CIM Requirements

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

384 ##### 6.3.2.1.3 Behavior Requirements

385 ###### 6.3.2.1.3.1 Preconditions

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

6.3.2.1.3.2 Pseudo Code

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

6.3.2.2 Show Command Form for a Single Instance – CIM_USBRedirectionSAP 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_HostedAccessPoint where only one reference is specified and the reference is to an instance of CIM_USBRedirectionSAP.

6.3.2.2.1 Command Form

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

6.3.2.2.2 CIM Requirements

See CIM_HostedAccessPoint in the “CIM Elements” section of the *USB Redirection 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_USBRedirectionSAP, which is referenced by CIM_HostedAccessPoint.

6.3.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_HostedAccessPoint”, $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 the `show` command issued against CIM_HostedAccessPoint where both references are specified; therefore, the desired instance is unambiguously identified.

6.3.2.3.1 Command Form

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

6.3.2.3.2 CIM Requirements

See CIM_HostedAccessPoint in the “CIM Elements” section of the *USB Redirection 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_ComputerSystem through CIM_HostedAccessPoint association.

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

6.3.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_HostedAccessPoint”, $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 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 mappings are 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 Multiple Instances – CIM_ComputerSystem Reference

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

6.4.2.1.1 Command Form

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

6.4.2.1.2 CIM Requirements

See CIM_ServiceAffectsElement in the “CIM Elements” section of the *USB Redirection 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_ComputerSystem, 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_USBDevice Reference

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

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 *USB Redirection 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_USBDevice, 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 Multiple Instances – CIM_USBRedirectionService Reference

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

6.4.2.3.1 Command Form

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

6.4.2.3.2 CIM Requirements

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

6.4.2.3.3 Behavior Requirements

6.4.2.3.3.1 Preconditions

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

6.4.2.3.3.2 Pseudo Code

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

6.4.2.4 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; therefore, the desired instance is unambiguously identified.

6.4.2.4.1 Command Form

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

6.4.2.4.2 CIM Requirements

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

6.4.2.4.3 Behavior Requirements

6.4.2.4.3.1 Preconditions

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

`$instanceB` represents the instance of CIM_ComputerSystem or CIM_USBDevice which is referenced by CIM_ServiceAffectsElement.

6.4.2.4.3.2 Pseudo Code

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

6.5 CIM_SAPAvailableForElement

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 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.

521 **Table 5 – Command Verb Requirements for CIM_SAPAvailableForElement**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.5.2.
start	Not supported	
stop	Not supported	

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

524 ### 6.5.1 Ordering of Results

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

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

528 ### 6.5.2 Show

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

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

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

536 ##### 6.5.2.1.1 Command Form

537 **`show <CIM_SAPAvailabileToElement multiple instances>`**

538 ##### 6.5.2.1.2 CIM Requirements

539 See CIM_SAPAvailableToElement in the “CIM Elements” section of the *USB Redirection Profile* for the
540 list of mandatory properties.

541 ##### 6.5.2.1.3 Behavior Requirements

542 ###### 6.5.2.1.3.1 Preconditions

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

6.5.2.1.3.2 Pseudo Code

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

6.5.2.2 Show Command Form for a Single Instance – CIM_USBDevice Reference

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

6.5.2.2.1 Command Form

```
show <CIM_SAPAvailabileToElement single instance>
```

6.5.2.2.2 CIM Requirements

See CIM_SAPAvailableToElement in the “CIM Elements” section of the *USB Redirection 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_USBDevice, which is referenced by CIM_SAPAvailabileToElement.

6.5.2.2.3.2 Pseudo Code

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

6.5.2.3 Show Command Form for Multiple Instances – CIM_USBRedirectionSAP Reference

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

6.5.2.3.1 Command Form

```
show <CIM_SAPAvailabileToElement multiple instances>
```

6.5.2.3.2 CIM Requirements

See CIM_SAPAvailableToElement in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.5.2.3.3 Behavior Requirements

6.5.2.3.3.1 Preconditions

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

6.5.2.3.3.2 Pseudo Code

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

6.5.2.4 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_SAPAvailableForElement where both references are specified; therefore, the desired instance is unambiguously identified.

6.5.2.4.1 Command Form

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

6.5.2.4.2 CIM Requirements

See CIM_SAPAvailableToElement in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.5.2.4.3 Behavior Requirements

6.5.2.4.3.1 Preconditions

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

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

6.5.2.4.3.2 Pseudo Code

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

6.6 CIM_ServiceAccessBySAP

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 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_ServiceAccessBySAP

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

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

No mappings are 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_ServiceAccessBySAP, implementations shall utilize the following algorithm to produce the natural (that is, default) ordering:

- Results for CIM_ServiceAccessBySAP 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_ServiceAccessBySAP. Implementations shall support the use of the `show` verb with CIM_ServiceAccessBySAP.

6.6.2.1 Show Command Form for a Single Instance – CIM_USBRedirectionSAP Reference

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

6.6.2.1.1 Command Form

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

6.6.2.1.2 CIM Requirements

See CIM_ServiceAccessBySAP in the “CIM Elements” section of the *USB Redirection 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_USBRedirectionSAP, which is referenced by CIM_ServiceAccessBySAP.

6.6.2.1.3.2 Pseudo Code

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

6.6.2.2 Show Command Form for Multiple Instances – CIM_USBRedirectionService 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_ServiceAccessBySAP where only one reference is specified and the reference is to an instance of CIM_USBRedirectionService.

6.6.2.2.1 Command Form

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

6.6.2.2.2 CIM Requirements

See CIM_ServiceAccessBySAP in the “CIM Elements” section of the *USB Redirection 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_USBRedirectionService, which is referenced by CIM_ServiceAccessBySAP.

6.6.2.2.3.2 Pseudo Code

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

6.6.2.3 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` command issued against CIM_ServiceAccessBySAP where both references are specified; and therefore, the desired instance is unambiguously identified.

6.6.2.3.1 Command Form

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

6.6.2.3.2 CIM Requirements

See CIM_ServiceAccessBySAP in the “CIM Elements” section of the *USB Redirection 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_USBRedirectionService through CIM_ServiceAccessBySAP association.

`$instanceB` represents the other instance of CIM_USBRedirectionSAP which is referenced by CIM_ServiceAccessBySAP.

6.6.2.3.3.2 Pseudo Code

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

6.7 CIM_RemoteAccessAvailableToElement

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 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

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

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

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.7.2.
start	Not supported	
stop	Not supported	

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

681 ### 6.7.1 Ordering of Results

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

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

686 ### 6.7.2 Show

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

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

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

695 ##### 6.7.2.1.1 Command Form

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

697 ##### 6.7.2.1.2 CIM Requirements

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

6.7.2.1.3 Behavior Requirements

6.7.2.1.3.1 Preconditions

`$instance` represents the instance of CIM_RemoteServiceAccessPoint, which is referenced by CIM_RemoteAccessAvailableToElement.

6.7.2.1.3.2 Pseudo Code

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

6.7.2.2 Show Command Form for a Single Instance – CIM_USBRedirectionSAP 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_RemoteAccessAvailableToElement where only one reference is specified and the reference is to an instance of CIM_USBRedirectionSAP.

6.7.2.2.1 Command Form

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

6.7.2.2.2 CIM Requirements

See CIM_RemoteAccessAvailableToElement in the “CIM Elements” section of the *USB Redirection 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_USBRedirectionSAP, which is referenced by CIM_RemoteAccessAvailableToElement.

6.7.2.2.3.2 Pseudo Code

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

6.7.2.3 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` command issued against CIM_RemoteAccessAvailableToElement where both references are specified; therefore, the desired instance is unambiguously identified.

6.7.2.3.1 Command Form

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

6.7.2.3.2 CIM Requirements

See CIM_RemoteAccessAvailableToElement in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.7.2.3.3 Behavior Requirements

6.7.2.3.3.1 Preconditions

`$instanceA` represents the referenced instance of CIM_RemoteServiceAccessPoint through CIM_RemoteAccessAvailableToElement association.

`$instanceB` represents the other instance of CIM_USBRedirectionSAP which is referenced by CIM_RemoteAccessAvailableToElement.

6.7.2.3.3.2 Pseudo Code

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

6.8 CIM_BindsTo

The `cd` and `help` 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 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_BindsTo

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.8.2.
start	Not supported	
stop	Not supported	

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

6.8.1 Ordering of Results

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

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

6.8.2 Show

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

6.8.2.1 Show Command Form for Multiple Instances – CIM_ProtocolEndpoint Reference

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

6.8.2.1.1 Command Form

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

6.8.2.1.2 CIM Requirements

See CIM_BindsTo in the "CIM Elements" section of the *USB Redirection Profile* for the list of mandatory properties.

6.8.2.1.3 Behavior Requirements

6.8.2.1.3.1 Preconditions

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

6.8.2.1.3.2 Pseudo Code

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

6.8.2.2 Show Command Form for a Single Instance – CIM_USBRedirectionSAP Reference

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

6.8.2.2.1 Command Form

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

6.8.2.2.2 CIM Requirements

See CIM_BindsTo in the "CIM Elements" section of the *USB Redirection Profile* for the list of mandatory properties.

6.8.2.2.3 Behavior Requirements

6.8.2.2.3.1 Preconditions

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

6.8.2.2.3.2 Pseudo Code

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

6.8.2.3 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` command issued against CIM_BindsTo where both references are specified; therefore, the desired instance is unambiguously identified.

6.8.2.3.1 Command Form

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

6.8.2.3.2 CIM Requirements

See CIM_BindsTo in the "CIM Elements" section of the *USB Redirection Profile* for the list of mandatory properties.

6.8.2.3.3 Behavior Requirements

6.8.2.3.3.1 Preconditions

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

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

6.8.2.3.3.2 Pseudo Code

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

6.9 CIM_LogicalIdentity

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 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_LogicalIdentity

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

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

6.9.1 Ordering of Results

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

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

6.9.2 Show

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

6.9.2.1 Show Command Form for a Single Instance – CIM_USBDevice 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_LogicalIdentity where only one reference is specified and the reference is to an instance of CIM_USBDevice.

6.9.2.1.1 Command Form

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

6.9.2.1.2 CIM Requirements

See CIM_LogicalIdentity in the "CIM Elements" section of the *USB Redirection Profile* for the list of mandatory properties.

6.9.2.1.3 Behavior Requirements

6.9.2.1.3.1 Preconditions

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

6.9.2.1.3.2 Pseudo Code

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

6.9.2.2 Show Command Form for a Single Instance – A Concrete Subclass of CIM_LogicalDevice 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_LogicalIdentity where only one reference is specified and the reference is to an instance of a concrete subclass of CIM_LogicalDevice.

6.9.2.2.1 Command Form

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

6.9.2.2.2 CIM Requirements

See CIM_LogicalIdentity in the "CIM Elements" section of the *USB Redirection 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 a concrete subclass of CIM_LogicalDevice, which is referenced by CIM_LogicalIdentity.

6.9.2.2.3.2 Pseudo Code

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

6.9.2.3 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` command issued against instances of CIM_LogicalIdentity where both references are specified; therefore, the desired instance is unambiguously identified.

6.9.2.3.1 Command Form

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

6.9.2.3.2 CIM Requirements

See CIM_LogicalIdentity in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.9.2.3.3 Behavior Requirements

6.9.2.3.3.1 Preconditions

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

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

6.9.2.3.3.2 Pseudo Code

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

6.10 CIM_USBDevice

The `cd` and `help` 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 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.

884 **Table 10 – Command Verb Requirements for CIM_USBDevice**

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

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

887 ### 6.10.1 Ordering of Results

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

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

891 ### 6.10.2 Create

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

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

895 #### 6.10.2.1 Create Specifying Properties

896 ##### 6.10.2.1.1 Command Form

897 **`create <CIM_USBDevice> <propertyname1>=<propertyvalue1>`**
898 **`   <propertyname`*`n`*`>=<propertyvalue`*`n`*`>`**

899 ##### 6.10.2.1.2 CIM Requirements

900 See CIM_USBRedirectionService.CreateUSBDevice in the “CIM Elements” section of the *USB*
901 *Redirection Profile* for the list of mandatory properties.

902 ##### 6.10.2.1.3 Behavior Requirements

903 ###### 6.10.2.1.3.1 Preconditions

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

6.10.2.1.3.2 Pseudo Code

```
//1 Find the Service for the system to create the USB Device
#Error = &smOpAssociators(
       $system.getObjectPath(),
       “CIM_HostedService”,
       “CIM_USBRedirectionService”,
       NULL,
       NULL,
       NULL,
       $SvcInstancePaths[])
if (0 != #Error.code)
   {
   &smProcessOpError (#Error);
   //includes &smEnd;
   }
$service = $SvcInstancePaths[0];
//2 populate the properties
$USBDeviceTemplate = smNewInstance (“CIM_USBDevice”);
for #i < n
   {
   $USBDeviceTemplate.<propertname#i> = <propertyvalue#i>;
   }
//3 invoke the method
$USBDevice = smNewInstance (“CIM_USBDevice”);
//build the parameter lists and invoke the method
%InArguments[] = {newArgument(“NewUSBDevice”, $USBDeviceTemplate)}
%OutArguments[] = { newArgument(“USBDevice”,
                    $USBDevice.GetObjectPath()) };
//invoke method
#returnStatus = smOpInvokeMethod ($service.GetObjectPath(),
   “CreateUSBDevice”,
   %InArguments[],
   %OutArguments[]);
//3 process return code to CLP Command Status
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;
   }
}
//if CIM op failed
else if (0 == #returnStatus)
   {
   //completed successfully, show created instance
   &smShowInstance($USBDevice.getObjectPath(), NULL);
   &smEnd;
   }
else if (4 == #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
   {
   //invalid parameter
   $OperationError = smNewInstance(“CIM_Error”);
   //CIM_ERR_FAILED
   $OperationError.CIMStatusCode = 1;
   //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;
   }
```

6.10.3 Delete

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

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

6.10.3.1 Delete a Single Instance

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

6.10.3.1.1 Command Form

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

6.10.3.1.2 CIM Requirements

See CIM_USBDevice in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.10.3.1.3 Behavior Requirements

```
$instance=<CIM_USBDevice single instance>
&smOpDeleteInstance ( $instance.GetObjectPath() );
&smEnd;
```

6.10.4 Show

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

6.10.4.1 Show a Single Instance

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

6.10.4.1.1 Command Form

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

6.10.4.1.2 CIM Requirements

See CIM_USBDevice in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.10.4.1.3 Behavior Requirements

6.10.4.1.3.1 Preconditions

`$instance` represents the instance of CIM_USBDevice.

`#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.

6.10.4.1.3.2 Pseudo Code

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

6.10.4.2 Show Multiple Instances – CIM_ServiceAvailableForElement

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

6.10.4.2.1 Command Form

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

6.10.4.2.2 CIM Requirements

See CIM_USBDevice in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.10.4.2.3 Behavior Requirements

6.10.4.2.3.1 Preconditions

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

6.10.4.2.3.2 Pseudo Code

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

6.10.4.3 Show Multiple Instances – CIM_ServiceAffectsElement

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

6.10.4.3.1 Command Form

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

6.10.4.3.2 CIM Requirements

See CIM_USBDevice in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.10.4.3.3 Behavior Requirements

6.10.4.3.3.1 Preconditions

`$containerInstance` represents the instance of CIM_USBRedirectionService for which the scoped instance of CIM_USBDevice is being displayed. The CIM_USBRedirectionService is associated to targeted instances of CIM_USBDevice via a CIM_ServiceAvailableForElement 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.

6.10.4.3.3.2 Pseudo Code

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

6.11 CIM_RemoteServiceAccessPoint

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 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.

1112 **Table 11 – Command Verb Requirements for CIM_RemoteServiceAccessPoint**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
Set	May	See 6.11.2.
show	Shall	See 6.11.3.
start	Not supported	
stop	Not supported	

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

1115 ### 6.11.1 Ordering of Results

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

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

1119 ### 6.11.2 Set

1120 #### 6.11.2.1 General Usage of Set for a Single Property

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

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

1126 ##### 6.11.2.1.1 Command Form

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

1128 ##### 6.11.2.1.2 CIM Requirements

1129 See CIM_RemoteServiceAccessPoint in the “CIM Elements” section of the *USB Redirection Profile* for
1130 the list of mandatory properties.

1131 ##### 6.11.2.1.3 Behavior Requirements

1132 ###### 6.11.2.1.3.1 Pseudo Code

1133 ```
$instance=<CIM_RemoteServiceAccessPoint single object>
1134 #propertyNames[] = <propertname>
1135 #propertyValues[] = <propertyvalue>
1136 &smSetInstance ( $instance, #propertyNames, #propertyValues );
1137 &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 *USB Redirection Profile*.

6.11.2.2.1 Command Form

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

6.11.2.2.2 CIM Requirements

See CIM_RemoteServiceAccessPoint in the “CIM Elements” section of the *USB Redirection 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_RemoteServiceAccessPoint.

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_RemoteServiceAccessPoint. Implementations shall support the use of the `show` verb with CIM_RemoteServiceAccessPoint.

6.11.3.1 Show a Single Instance

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

6.11.3.1.1 Command Form

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

6.11.3.1.2 CIM Requirements

See CIM_RemoteServiceAccessPoint in the “CIM Elements” section of the *USB Redirection 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_RemoteServiceAccessPoint.

`#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 – CIM_RemoteAccessAvailableToElement

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

6.11.3.2.1 Command Form

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

6.11.3.2.2 CIM Requirements

See CIM_RemoteServiceAccessPoint in the "CIM Elements" section of the *USB Redirection 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_USBRedirectionSAP for which the scoped instance of CIM_RemoteServiceAccessPoint is being displayed. The CIM_USBRedirectionSAP is associated to targeted instances of CIM_RemoteServiceAccessPoint via a CIM_RemoteAccessAvailableToElement 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_RemoteServiceAccessPoint",
   "CIM_RemoteAccessAvailableToElement", $containerInstance.getObjectPath(),
   #propertylist[] );
&smEnd;
```

6.12 CIM_EnabledLogicalElementCapabilities

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 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

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

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

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

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

1214 ### 6.12.1 Ordering of Results

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

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

1219 ### 6.12.2 Show

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

1223 #### 6.12.2.1 Show a Single Instance

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

1226 ##### 6.12.2.1.1 Command Form

1227 **`show <CIM_EnabledLogicalElementCapabilities single instance>`**

1228 ##### 6.12.2.1.2 CIM Requirements

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

6.12.2.1.3 Behavior Requirements

6.12.2.1.3.1 Preconditions

`$instance` represents the instance of CIM_EnabledLogicalElementCapabilities.

`#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.

6.12.2.1.3.2 Pseudo Code

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

6.12.2.2 Show Multiple Instances

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

6.12.2.2.1 Command Form

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

6.12.2.2.2 CIM Requirements

See CIM_EnabledLogicalElementCapabilities in the "CIM Elements" section of the *USB Redirection 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_EnabledLogicalElementCapabilities through the CIM_MemberOfCollection 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.

6.12.2.2.3.2 Pseudo Code

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

6.13 CIM_USBRedirectionCapabilities

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.

1268 **Table 13 – Command Verb Requirements for CIM_USBRedirectionCapabilities**

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

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

1271 ### 6.13.1 Ordering of Results

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

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

1275 ### 6.13.2 Show

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

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

1281 #### 6.13.2.1 Show a Single Instance

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

1283 ##### 6.13.2.1.1 Command Form

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

1285 ##### 6.13.2.1.2 CIM Requirements

1286 See CIM_USBRedirectionCapabilities in the “CIM Elements” section of the *USB Redirection Profile* for the
1287 list of mandatory properties.

1288 ##### 6.13.2.1.3 Behavior Requirements

1289 ###### 6.13.2.1.3.1 Preconditions

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

6.13.2.1.3.2 Pseudo Code

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

6.13.2.2 Show Multiple Instances

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

6.13.2.2.1 Command Form

```
show <CIM_USBRedirectionCapabilities multiple instances>
```

6.13.2.2.2 CIM Requirements

See CIM_USBRedirectionCapabilities in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.13.2.2.3 Behavior Requirements

6.13.2.2.3.1 Preconditions

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

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

6.13.2.2.3.2 Pseudo Code

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

6.14 CIM_USBRedirectionSAP

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

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

1331 **Table 14 – Command Verb Requirements for CIM_USBRedirectionSAP**

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

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

1333 ### 6.14.1 Ordering of Results

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

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

1337 ### 6.14.2 Create

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

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

1342 #### 6.14.2.1 Create Specifying Properties

1343 ##### 6.14.2.1.1 Command Form

1344 ```
create <CIM_USBRedirectionSAP> <propertyname1>=<propertyvalue1>
1345   <propertyname n>=<propertyvalue n>
```

1346 ##### 6.14.2.1.2 CIM Requirements

1347 See CIM_USBRedirectionSAP in the “CIM Elements” section of the *USB Redirection Profile* for the list of
1348 mandatory properties.

1349 ##### 6.14.2.1.3 Behavior Requirements

1350 ###### 6.14.2.1.3.1 Preconditions

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

6.14.2.1.3.2 Pseudo Code

```
//1 Find the Service for the system to create the Redirection SAP
#Error = &smOpAssociators (
       $system.getObjectPath(),
       “CIM_HostedService”,
       “CIM_USBRedirectionService”,
       NULL,
       NULL,
       NULL,
       $SvcInstancePaths[] )
if (0 != #Error.code)
   {
   &smProcessOpError (#Error);
   //includes &smEnd;
   }
$service = $SvcInstancePaths[0];
//2 convert command line parms to method parms
$USBSAPTemplate = smNewInstance (“CIM_USBRedirectionSAP”);
#CreateDevices = FALSE;
$NewUSBDevice = smNewInstance (“CIM_USBDevice”);
$RemoteSAP = smNewInstance (“CIM_RemoteServiceAccessPoint”);
for #i < n
{
   if (<propertname#i> == “NewSAPRequestedStatesSupported”){
      #NewRequestedStatesSupported[] = <propertyvalue#i>;
   }
   else if (<propertname#i> == “NewUSBDeviceUSBVersion”)   {
      $NewUSBDevices.USBVersion = <propertyvalue#i>;
      #CreateDevices = TRUE;
   }
      else if (<propertname#i> == “NewUSBDeviceClassCode”) {
      $NewUSBDevices.ClassCode = <propertyvalue#i>;
      #CreateDevices = TRUE;
   }
   else if (<propertname#i> == “NewUSBDeviceSubclassCode”) {
      $NewUSBDevices.SubclassCode = <propertyvalue#i>;
      #CreateDevices = TRUE;
   }
   else if (<propertname#i> == “NewUSBDeviceCommandTimeout”)   {
      $NewUSBDevices.CommandTimeout = <propertyvalue#i>;
      #CreateDevices = TRUE;
   }
   else if (<propertname#i> == “NewUSBDeviceProtocolCode”) {
      $NewUSBDevices.ProtocolCode = <propertyvalue#i>;
      #CreateDevices = TRUE;
   }
   else if (<propertname#i> == “NewUSBDeviceMaxPacketSize”){
      $NewUSBDevices.MaxPacketSize = <propertyvalue#i>;
```

```
      #CreateDevices = TRUE;
   }
   else if (<propertname#i> == “NewUSBDeviceVendorID”)  {
      $NewUSBDevices.VendorID = <propertyvalue#i>;
      #CreateDevices = TRUE;
   }
   else if (<propertname#i> == “NewUSBDeviceProductID”) {
      $NewUSBDevices.ProductID = <propertyvalue#i>;
      #CreateDevices = TRUE;
   }
   else if (<propertname#i> == “NewUSBDeviceDeviceReleaseNumber”)  {
      $NewUSBDevices.DeviceReleaseNumber = <propertyvalue#i>;
      #CreateDevices = TRUE;
   }
   else if (<propertname#i> == “NewUSBDeviceSerialNumber”) {
      $NewUSBDevices.SerialNumber = <propertyvalue#i>;
      #CreateDevices = TRUE;
   }
   else if (<propertname#i> == “NewUSBDeviceManufacturer”) {
      $NewUSBDevices.Manufacturer = <propertyvalue#i>;
      #CreateDevices = TRUE;
   }
   else if (<propertname#i> == “NewUSBDeviceProduct”)   {
      $NewUSBDevices.Product = <propertyvalue#i>;
      #CreateDevices = TRUE;
   }
   else if (<propertname#i> == “RemoteSAPAccessContext”) {
   $RemoteSAP.AccessContext = <propertyvalue#i>;
   }
   else if (<propertname#i> == “RemoteSAPAccessInfo”) {
   $RemoteSAP.AccessInfo = <propertyvalue#i>;
   }
   else if (<propertname#i> == “RemoteSAPInfoFormat”) {
   $RemoteSAP.InfoFormat = <propertyvalue#i>;
   }
   else if (<propertname#i> == “RemoteSAPOtherAccessContext”) {
   $RemoteSAP.OtherAccessContext = <propertyvalue#i>;
   }
   else if (<propertname#i> == “RemoteSAPOtherInfoFormatDescription”) {
   $RemoteSAP.OtherInfoFormatDescription = <propertyvalue#i>;
   }
   else if (<propertname#i> == “USBDevices”) {
// #USBDevices[] contains references to instances of CIM_USBDevice produced
// from <propertyvalue#i> using conversion mechanism presumed to exist
   $USBDevices[]->= <propertyvalue#i>;
   }
   else {
      $USBSAPTemplate.<propertname#i> = <propertyvalue#i>;
   }
```

```
}
//3 invoke the method
$USBRedirectionSAP = smNewInstance (“CIM_USBRedirectionSAP”);
//build the parameter lists and invoke the method
If (#CreateDevices) {
// #NewUSBDevicesStrings contains the embedded instance produced from
// $NewUSBDevices using conversion mechanism presumed to exist
   #NewUSBDevicesStrings[0] = $NewUSBDevices;
}
%InArguments[] = { newArgument(“NewUSBRedirectionSAP”, $USBSAPTemplate),
                   newArgument(“NewSAPRequestedStatesSupported”,
                      #NewRequestedStatesSupported[]),
                      newArgument(“USBDevices”, $USBDevices[]->),
                      newArgument(“CreateDevices”, $CreateDevices),
                      newArgument(“NewUSBDevices”, #NewUSBDevicesStrings[]),
                      newArgument(“NewRemoteServiceAccessPoint”, $RemoteSAP) };
%OutArguments[] = { newArgument(“SAP”, $USBRedirectionSAP.GetObjectPath()) };
//invoke method
#returnStatus = smOpInvokeMethod ($service.GetObjectPath(),
   “CreateRedirectionSAP”,
   %InArguments[],
   %OutArguments[]);
//3 process return code to CLP Command Status
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;
   }
}//if CIM op failed
else if (0 == #returnStatus)  {
   //completed successfully, show created instance
   &smShowInstance($USBRedirectionSAP.getObjectPath(), NULL);
&smEnd;
}
else if (4 == #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   {
   //invalid parameter
   $OperationError = smNewInstance("CIM_Error");
   //CIM_ERR_FAILED
   $OperationError.CIMStatusCode = 1;
   //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;
}
```

6.14.3 Delete

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

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

6.14.3.1 Delete a Single Instance

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

6.14.3.1.1 Command Form

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

6.14.3.1.2 CIM Requirements

See CIM_USBRedirectionSAP in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.14.3.1.3 Behavior Requirements

6.14.3.1.3.1 Preconditions

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

6.14.3.1.3.2 Pseudo Code

```
//1 Find the Service for the system to delete the Redirection SAP
#Error = &smOpAssociators(
       $system.getObjectPath(),
       “CIM_HostedService”,
       “CIM_USBRedirectionService”,
       NULL,
       NULL,
       NULL,
       $SvcInstancePaths[])
if (0 != #Error.code)
   {
   &smProcessOpError (#Error);
   //includes &smEnd;
   }
$service = $SvcInstancePaths[0];
//2 convert command line parms to method parms
$instance=<CIM_RedirectionSAP single instance>;
```

```
// Presumably the Delete Redirection SAP operation should always
// delete all the devices created by the Create Redirection SAP
// operations. So the DeleteUSBDevices parameter to the
// DeleteRedirectionSAP extrinsic is always set to TRUE
$DeleteDevices = TRUE;
//3 invoke the method
//build the parameter lists and invoke the method
%InArguments[] = {newArgument(“RedirectionSAP”, $instance),
                  newArgument(“DeleteUSBDevices”, $DeleteDevices)};
//invoke method
#returnStatus = smOpInvokeMethod ($service.GetObjectPath(),
                  “DeleteRedirectionSAP”,
                  %InArguments[],
                  %OutArguments[]);
//3 process return code to CLP Command Status
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;
   }
}//if CIM op failed
else if (0 == #returnStatus)  {
   //completed successfully
}
else if (4 == #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   {
   //invalid parameter
   $OperationError = smNewInstance(“CIM_Error”);
   //CIM_ERR_FAILED
   $OperationError.CIMStatusCode = 1;
   //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;
}
```

6.14.4 Set

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

6.14.4.1 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 *USB Redirection Profile*.

6.14.4.1.1 Command Form

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

6.14.4.1.2 CIM Requirements

See CIM_USBRedirectionSAP in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.14.4.1.3 Behavior Requirements

6.14.4.1.3.1 Pseudo Code

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

6.14.4.2 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 *USB Redirection Profile*.

6.14.4.2.1 Command Form

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

6.14.4.2.2 CIM Requirements

See CIM_USBRedirectionSAP in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.14.4.2.3 Behavior Requirements

6.14.4.2.3.1 Preconditions

`$instance` represents the instance of CIM_USBRedirectionSAP.

6.14.4.2.3.2 Pseudo Code

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

6.14.5 Show

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

6.14.5.1 Show a Single Instance

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

6.14.5.1.1 Command Form

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

6.14.5.1.2 CIM Requirements

See CIM_USBRedirectionSAP in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.14.5.1.3 Behavior Requirements

6.14.5.1.3.1 Preconditions

`$instance` represents the instance of CIM_USBRedirectionSAP.

`#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.

6.14.5.1.3.2 Pseudo Code

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

6.14.5.2 Show Multiple Instances – CIM_HostedAccessPoint

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

6.14.5.2.1 Command Form

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

6.14.5.2.2 CIM Requirements

See CIM_USBRedirectionSAP in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.14.5.2.3 Behavior Requirements

6.14.5.2.3.1 Preconditions

`$containerInstance` represents the instance of CIM_ComputerSystem for which the scoped instance of CIM_USBRedirectionSAP is being displayed. The CIM_USBRedirectionSAP is associated to CIM_ComputerSystem via a CIM_HostedAccessPoint 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.

6.14.5.2.3.2 Pseudo Code

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

6.14.5.3 Show Multiple Instances – CIM_SAPAvailableForElement

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

6.14.5.3.1 Command Form

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

6.14.5.3.2 CIM Requirements

See CIM_USBRedirectionSAP in the "CIM Elements" section of the *USB Redirection Profile* for the list of mandatory properties.

6.14.5.3.3 Behavior Requirements

6.14.5.3.3.1 Preconditions

`$containerInstance` represents the instance of CIM_ComputerSystem for which the scoped instance of CIM_USBRedirectionSAP is being displayed. The CIM_USBRedirectionSAP is associated to CIM_ComputerSystem via a CIM_SAPAvailableForElement 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.

6.14.5.3.3.2 Pseudo Code

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

6.14.5.4 Show Multiple Instances – CIM_ServiceAccessBySAP

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

6.14.5.4.1 Command Form

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

6.14.5.4.2 CIM Requirements

See CIM_USBRedirectionSAP in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.14.5.4.3 Behavior Requirements

6.14.5.4.3.1 Preconditions

`$containerInstance` represents the instance of CIM_USBRedirectionService for which the scoped instance of CIM_USBRedirectionSAP is being displayed. The CIM_USBRedirectionSAP is associated to CIM_USBRedirectionService via a CIM_ServiceAccessBySAP 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.

6.14.5.4.3.2 Pseudo Code

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

6.14.6 Start

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

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

6.14.6.1 Start a Single Instance

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

6.14.6.1.1 Command Form

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

6.14.6.1.2 CIM Requirements

See CIM_USBRedirectionSAP in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.14.6.1.3 Behavior Requirements

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

6.14.7 Stop

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

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

6.14.7.1 Stop a Single Instance

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

6.14.7.1.1 Command Form

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

6.14.7.1.2 CIM Requirements

See CIM_USBRedirectionSAP in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.14.7.1.3 Behavior Requirements

6.14.7.1.3.1 Preconditions

`$instance` represents the targeted instance of CIM_USBRedirectionSAP.

6.14.7.1.3.2 Pseudo Code

```
$instance=<CIM_USBRedirectionSAP single instance>
&smRequestStateChange ( $instance.getObjectPath(), “Offline” );
&smEnd;
```

6.14.7.2 Set RequestedState to “Disabled”

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

6.14.7.2.1 Command Form

```
set <CIM_USBRedirectionSAP single instance> RequestedState=“Disabled”
```

6.14.7.2.2 CIM Requirements

See CIM_USBRedirectionSAP in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.14.7.2.3 Behavior Requirements

6.14.7.2.3.1 Preconditions

`$instance` represents the targeted instance of CIM_USBRedirectionSAP.

6.14.7.2.3.2 Pseudo Code

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

6.15 CIM_USBRedirectionService

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

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

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

Table 15 – Command Verb Requirements for CIM_USBRedirectionService

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

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

6.15.1 Ordering of Results

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

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

6.15.2 Set

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

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

6.15.2.1 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 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 *USB Redirection Profile*.

6.15.2.1.1 Command Form

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

6.15.2.1.2 CIM Requirements

See CIM_USBRedirectionService in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.15.2.1.3 Behavior Requirements

6.15.2.1.3.1 Pseudo Code

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

6.15.2.2 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 *USB Redirection Profile*.

6.15.2.2.1 Command Form

```
set <CIM_USBRedirectionService multiple objects> <propertyname1>=<propertyvalue1>
   <propertynameN>=<propertyvalueN>
```

6.15.2.2.2 CIM Requirements

See CIM_USBRedirectionService in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.15.2.2.3 Behavior Requirements

6.15.2.2.3.1 Preconditions

`$instance` represents the instance of CIM_USBRedirectionService.

6.15.2.2.3.2 Pseudo Code

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

6.15.3 Show

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

6.15.3.1 Show Command Form for a Single Instance

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

6.15.3.1.1 Command Form

```
show <CIM_USBRedirectionService single instance>
```

6.15.3.1.2 CIM Requirements

See CIM_USBRedirectionService in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.15.3.1.3 Behavior Requirements

6.15.3.1.3.1 Preconditions

`$instance` represents the instance of CIM_USBRedirectionService.

`#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.

6.15.3.1.3.2 Pseudo Code

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

6.15.4 Start

6.15.4.1 Start Command Form for a Single Instance

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

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

6.15.4.1.1 Command Form

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

6.15.4.1.2 CIM Requirements

See CIM_USBRedirectionService in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.15.4.1.3 Behavior Requirements

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

6.15.5 Stop

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

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

6.15.5.1 Stop Command Form for a Single Instance

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

6.15.5.1.1 Command Form

```
stop <CIM_USBRedirectionService single instance>
```

6.15.5.1.2 CIM Requirements

See CIM_USBRedirectionService in the “CIM Elements” section of the *USB Redirection Profile* for the list of mandatory properties.

6.15.5.1.3 Behavior Requirements

```
$instance=<CIM_USBRedirectionService single instance>;
&smStopRSC ( $instance.getObjectPath() );
&smEnd;
```

ANNEX A
(informative)

Change Log

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