



1  
2  
3  
4

**Document Number: DSP0214**

**Date: 2007-03-07**

**Version: 1.0.2**

5 **Server Management Command Line Protocol**  
6 **(SM CLP) Specification**

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

## Server Management Command Line Protocol (SM CLP) Specification

10 Copyright notice

11 Copyright © 2007 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

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

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27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party  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

CONTENTS

31 Foreword ..... 5

32 Introduction ..... 6

33 Server Management Command Line Protocol (SM CLP) Specification ..... 11

34 1 Scope ..... 11

35 2 Normative References..... 11

36 3 Terms and Definitions ..... 12

37 4 Symbols and Abbreviated Terms ..... 16

38 5 Server Management Command Line Protocol (SM CLP) Specification ..... 18

39 5.1 Semantics ..... 18

40 5.2 Syntax ..... 28

41 6 SM CLP Verbs..... 51

42 6.1 Requirements for Supporting CVS ..... **Error! Bookmark not defined.**

43 6.2 cd ..... 52

44 6.3 create ..... 56

45 6.4 delete ..... 60

46 6.5 dump ..... 66

47 6.6 exit..... 71

48 6.7 help ..... 73

49 6.8 load ..... 75

50 6.9 reset ..... 78

51 6.10 set ..... 81

52 6.11 show..... 89

53 6.12 start ..... 117

54 6.13 stop ..... 120

55 6.14 version ..... 124

56 7 Standard Command Options..... 126

57 7.1 all..... 129

58 7.2 destination..... 129

59 7.3 display ..... 130

60 7.4 examine ..... 132

61 7.5 force ..... 133

62 7.6 help ..... 133

63 7.7 keep ..... 134

64 7.8 level..... 134

65 7.9 output ..... 135

66 7.10 source ..... 140

67 7.11 version ..... 140

68 7.12 wait..... 141

69 8 SM CLP Session ..... 141

70 8.1 Authentication, Authorization, and Audit..... 141

71 8.2 CLP Session ..... 142

72 8.3 Transport..... 148

73 Annex A (normative) SM CLP Command Grammar in ABNF Notation (RFC2234)..... 150

74 Annex B (informative) W3C Universal Resource Identifiers (URI)..... 156

75 Annex C (informative) W3C Extensible Markup Language (XML) ..... 157

76 Annex D (informative) POSIX Utility Conventions ..... 158

77 Annex E (informative) Conventions ..... 160

78 Annex F (informative) Notation ..... 161

## Server Management Command Line Protocol (SM CLP) Specification

79	Annex G (informative) Change History .....	162
80	Annex H (informative) Acknowledgements .....	163
81	Annex I (informative) Bibliography .....	164

82

### 83 **Figures**

84	Figure 1 – Session Establishment Sequence .....	144
85	Figure 2 – Command Interaction Sequences .....	145
86	Figure 3 – Session Switching Sequences.....	146
87	Figure 4 – Session Termination Sequences.....	147

88

### 89 **Tables**

90	Table 1 – CLP Reserved Characters and Character Sequences .....	28
91	Table 2 – Common Output Keywords.....	36
92	Table 3 – Command Status Keywords .....	37
93	Table 4 – Command Status Values and Tags .....	37
94	Table 5 – Message Keywords.....	37
95	Table 6 – Processing Error Values and Tags .....	38
96	Table 7 – Job Error Keywords .....	39
97	Table 8 – Error Type Values and Descriptions .....	39
98	Table 9 – CIM Status Code Values and Descriptions.....	40
99	Table 10 – Severity Values and Descriptions .....	41
100	Table 11 – Probable Cause Values and Descriptions .....	42
101	Table 12 – Verb Support Requirements .....	51
102	Table 13 – Data Element Types and all Option .....	92
103	Table 14 – Standard Command Options .....	127
104	Table 15 – Verb and Option Support .....	128
105	Table 16 – Output Option Arguments .....	135
106	Table 17 – Output Options for Controlling Command Status Output .....	140
107	Table 18 – Command Authorizations for CLP Groups .....	142
108	Table 19 – Telnet Transport Support Requirements .....	148

109

110

## Foreword

111 The *Server Management Command Line Protocol (SM CLP) Specification* (DSP0214) was prepared by  
112 the Server Management Working Group. This document was prepared in accordance with *ISO/IEC*  
113 *Directives, Part 2: Rules for the structure and drafting of International Standards*.

114 The *Server Management Command Line Protocol (SM CLP) Specification* specifies a common command  
115 line syntax and message protocol semantics for managing computer resources in Internet, enterprise,  
116 and service provider environments.

117 DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems  
118 management and interoperability.

119

### Introduction

120 This section contains an overview of the Command Line Protocol (CLP). This includes the goals behind  
121 creating the CLP and the specific problems that it attempts to resolve. In addition, this section lays the  
122 groundwork for the sections that follow by detailing the background and assumptions of the CLP. This  
123 includes the architecture assumed in the design of the CLP and the components within that architecture.

### 124 **Problem Statement**

125 The fundamental problem that is the impetus behind this specification is the growing need to rely on multi-  
126 vendor, out-of-band hardware and software management solutions as core components of an  
127 interoperable, heterogeneous, enterprise-wide management solution. By extending the DMTF  
128 specifications to include a CIM-based command line protocol for managing systems and devices, the  
129 DMTF comes closer to realizing its vision of enabling end-to-end, multi-vendor interoperability in  
130 management systems.

### 131 **Principal Goals**

132 The principal goal of this specification is to define a light-weight, human-oriented command line protocol  
133 that is also suitable for scripting environments. This includes a direct mapping to a subset of the CIM  
134 Schema. The command line protocol specifies the syntax and semantics used to allow the manipulation  
135 of the Managed Elements and Associations within servers, as collections or individually.

### 136 **Solution**

137 The solution proposed in this document is a command line protocol (CLP), which is transmitted and  
138 received over a text message-based transport protocol. The CLP is defined as a character-based  
139 message protocol and not as an interface, in a fashion similar to *Simple Mail Transfer Protocol*  
140 (RFC2821).

141 The CLP is a command/response protocol, which means that a text command message is transmitted  
142 from the Client over the transport protocol to the Manageability Access Point (MAP). The MAP receives  
143 the command and processes it. A text response message is then transmitted from the MAP back to the  
144 Client.

145 The CLP is designed to work over existing character-oriented transports. The specification contains  
146 mappings to Telnet and SSHv2, but any transport capable of carrying command/response message data  
147 of the type specified herein may be suitable for use as a transport.

148 The CLP enables internationalization by providing a mechanism for the Client to indicate to the MAP the  
149 language desired by the Client. Provided the MAP supports the requested language, output data will be  
150 presented to the user with the appropriate translations. This version of the CLP does not support specific  
151 internationalization of user account names and passwords because they can be in any specific language.  
152 In addition, the CLP input (commands and syntax) is not translated because CLP syntax is itself its own  
153 language.

154 The CLP allows for extensibility through four different mechanisms: verbs, options and option argument  
155 terms, command target terms, and target property terms. The conventions contained herein allow for  
156 implementers to extend the interface in a non-conflicting mechanism that allows for differentiation and  
157 experimentation without encroaching on the standard CLP syntax and semantics.

### 158 **General Syntax**

159 The general syntax for the CLP is of the form:

```
160 <verb> [<options>] [<target>] [<properties>]
```

161 The *verb* term refers to the specific command or action taking place. These are covered in detail in  
162 Clause 6. The list of verbs includes those that establish and retrieve data ("set" and "show"), create and  
163 remove records or instances ("create" and "delete"), change the state of a given target ("reset",  
164 "start", and "stop"), manage the current session ("cd", "version", and "exit"), and provide  
165 command information ("help").

166 The *target* term indicates the address or path of the target of the command. The format of this term is  
167 discussed in the *Server Management Managed Element (SM ME) Addressing Specification (DSP0215)*.  
168 This term can be an individual Managed Element, such as a disk, a NIC, the MAP itself, or even a  
169 service, such as the transport. This term can also be a collection of Managed Elements supported by the  
170 MAP, such as a system. It can also be an Association. There can be only one target term specified per  
171 command.

172 Command options always modify the action or behavior of the verb. Options may appear immediately  
173 after the verb on the Command Line and shall be preceded by a hyphen ("-"). They provide features such  
174 as changing the output format, allowing the command to apply to nested levels, requesting that the  
175 version of the command be displayed, and requesting help. Note that there may be zero or more option  
176 terms per command. For more information, see Clause 7.

177 Command target properties are attributes that may contain values associated with a target that are  
178 needed to process the command. Command target properties identify properties of the target's class that  
179 are to be retrieved or modified by the command. Valid command target property names are documented  
180 in the MOF file that defines the class. Implementations shall use the property name defined in the MOF  
181 file to identify the property of the class.

182 The properties themselves are manipulated with the commands in Clause 6. If a value is to be assigned  
183 to a property, the syntax shall be of the form "<property name>=<value>". There may be zero or more  
184 property terms per command.

### 185 **Architectural Assumptions**

186 There is an underlying assumption that the architecture the CLP is built upon, or is an interface into, is a  
187 CIM Server implementation. The CLP is organized around management tasks mapped to operations on  
188 CIM instances. It does this by retrieving or changing properties and invoking methods established in  
189 instances of the CIM Schema. The mapping of CLP commands to CIM elements is documented in the  
190 *CLP-to-CIM Mapping Specification (DSP0216)* to aid implementers and consumers of this specification.

191 The CLP consists of a set of specific functions intended for operational control of the server hardware and  
192 rudimentary control of the operating system. It is not intended to be a complete interface into managing  
193 the operating system. Therefore, the CLP contains the commands necessary to operate on a proper  
194 subset of the CIM Schema as defined in DSP0216.

195 The CLP is also architected to work over existing transports. It is assumed that the transports will provide  
196 the authentication and encryption necessary for the protocol. Role-based command use authorization is  
197 included in the CLP, but the architecture assumes that the CLP relies on the underlying transport for any  
198 access security and authentication. The CLP architecture is documented in the *SM Architecture White  
199 Paper (DSP2001)*.

### 200 **Architectural Concepts of the SM CLP**

201 The following sections describe some of the key concepts of the SM CLP. A detailed statement of the  
202 architecture of the SM CLP is available in DSP2001. An implementation of the CLP service is modeled  
203 using the *Command Line Protocol (CLP) Service Profile* (DSP1005).

### 204 **Physical Connection**

205 The physical connection and media between the Client and the MAP are outside the scope of this  
206 specification. Any physical connection that is capable of running one of the supported transports is  
207 assumed to be able to support the CLP. Because the supported transports themselves can run over IP, it  
208 is safe to assume that the CLP can be transmitted and received over any media or physical connection  
209 that supports IP. However, this does not limit support for the CLP to physical media or connections that  
210 support IP. In fact, it is reasonable for an implementation to be created in which the protocol never leaves  
211 the managed server.

### 212 **Transport Management**

213 This specification includes sections detailing the mapping of the CLP over two transport protocols: Telnet  
214 and SSHv2. The CLP is designed to be transport independent, but mappings to transports other than  
215 these two are outside the scope of this specification.

216 The Client is the terminus of one end of the connection; the CLP Service implementation is the other.  
217 CLP Service implementations manage the transport and the sessions that occur over the transport as  
218 required by DSP1005.

### 219 **Authentication, Authorization, and Encryption**

220 The CLP Service does not perform any authentication or encryption. It relies entirely on the transport to  
221 perform these functions. Session transport requirements are documented in 8.3.

222 To accommodate a single basis for user authorization, the user account database required by the  
223 transport is expected to share the user information with the CLP Service once the user is logged in. For  
224 more information, see 8.3.

225 The CLP Service authorizes commands through the use of authorization groups. Each CLP User shall be  
226 a member of at least one CLP Group. For more information, see 8.1.

227 The CLP contains commands for the creation, removal, and modification of user accounts, including  
228 authorization and access rights. For more information, see Clause 6.

### 229 **Sessions**

230 Sessions between a Client and a CLP Service are established over a transport protocol. After the session  
231 has been authenticated, the Client can begin to submit commands using the CLP Service. Each session  
232 has a unique context within the MAP. Within this context, the CLP Service keeps track of session  
233 characteristics. Implementations will maintain a session context and session characteristics as required  
234 by DSP1005. Examples of these characteristics include the Current Default Target, currently selected  
235 output mode, current output language, and the current user and session identifier. Commands for  
236 manipulating the session characteristics are included in the CLP. For more information, see 8.2.2.

### 237 **Input Editing**

238 The CLP is a command/response protocol. CLP implementations shall receive and parse an entire  
239 Command Line, complete with verb, command target term, options, and properties. The CLP Service  
240 shall not allow any interactive input or data editing. This does not preclude a vendor from providing such  
241 capability associated with the Client implementation, but any such capability is outside of the scope of this  
242 specification.



## Server Management Command Line Protocol (SM CLP) Specification

### 243 **Command Line Protocol Service**

244 The CLP Service is responsible for providing and enforcing the syntax and semantics of the CLP.  
245 Implementations will support being managed as required by DSP1005. This includes starting, stopping,  
246 and changing the attributes of the service.

### 247 **CLP Service Access Point**

248 The transport session is established to the CLP Service Access Point (SAP). The access point represents  
249 the physical and logical communication mechanism through which the CLP Service receives incoming  
250 connection requests. The CLP provides the mechanisms necessary to enable, disable, and configure the  
251 SAP. Implementations will support managing the supporting protocol stacks as required by DSP1005.

### 252 **Operation Management**

253 All commands submitted through the CLP Service create jobs within the MAP. There is one and only one  
254 global job queue within the MAP. Implementations shall track all jobs using this single job queue.

255 Operations follow the CIM\_Job schema, as defined in later sections. The CLP supports commands to  
256 query jobs, retrieve the interim status of jobs, retrieve the final status of jobs, and delete jobs. Operations  
257 are covered further in 5.1.6.

### 258 **Use Cases**

259 This section describes the intended features, functions, and uses of the CLP. Note that the CLP is not  
260 limited to these functions, but these are the specific uses for which the CLP was intended.

261 The CLP is designed to apply to a number of server topologies. This includes, but is not limited to, stand-  
262 alone servers, rack-mounted servers, blades, partitioned servers, and Telco servers. It is also suitable to  
263 manage any necessary enterprise components, enclosures, chassis, racks, and power supplies  
264 necessary to utilize servers.

265 The CLP provides the ability to enumerate and configure server hardware. This includes discovery of the  
266 current hardware configuration and properties, system settings, and local IO devices. The CLP provides  
267 some amount of configuration for local disk drives, including local arrays. The intention of providing this  
268 support is to allow initial logical unit creation for installation, provisioning, or both. It is not intended that  
269 the CLP Service be the primary interface for managing mass storage, because these standards and  
270 access points exist in the industry.

271 The CLP also includes the ability to select, control, and initiate the transfer of images. Also provided is the  
272 ability to control the boot configuration of any supported server. In addition, support for heartbeat and  
273 operating-system-status information is included.

274 Server state control is included in the CLP. This includes power control, intervention capability (to halt,  
275 reset, or shut down a server), and mechanisms to initiate a dump of the operating system.

276 Access to some system resources is also included in the CLP. This includes access and manipulation of  
277 any accessible logs; the ability to view and set remote status displays, LEDs, and alarms; the ability to  
278 configure alert destinations; and the ability to initiate a session with a remote text-based console device.

279 The CLP also supports normal expected user session functions such as help, version information, and  
280 the ability to exit or terminate a session.

### 281 **Known Limitations**

282 First and foremost, while CLP commands are mapped to CIM methods and operations, the CLP is not  
283 intended to be a complete mapping to every CIM method, property, or operation. The CLP supports a  
284 sufficient subset of CIM Server features to enable the CLP to be the primary locus of interaction for server  
285 management, regardless of server type, physical connection, or operating system state.

286 Another known limitation pertains to the intended Client. The CLP is primarily focused on an interactive  
287 experience with a human user or simple script. It is not intended to be the primary interface for advanced  
288 server management software to use to manage hardware. Consequently, the format of the commands  
289 and their responses, as well as the CIM methods, properties, and operations supported, are not always  
290 sufficient for the CLP Service Access Point to be the primary interface for advanced server management  
291 software.

## 292 Server Management Command Line Protocol (SM CLP) Specification

### 293 1 Scope

294 This document lays out the general framework for the Server Management Command Line Protocol (SM  
295 CLP). This specification is intended to guide developers of implementations of the SM CLP and optionally  
296 be used as a reference by system administrators and other users of SM CLP implementations.

297 The following subjects are within the scope of this document:

- 298 • Command Line Protocol syntax and semantics
- 299 • input format and output format
- 300 • accessing and traversing the target address space
- 301 • error handling and semantics
- 302 • session management, including mapping to supported transports
- 303 • session characteristics
- 304 • operation processing and reporting

305 The following subjects are outside the scope of this document:

- 306 • control command verbs, such as loop control, conditionals, or prompting
- 307 • regular expressions, such as mathematical or logical expressions
- 308 • command editor environment
- 309 • Client's shell environment
- 310 • physical interconnects
- 311 • complex data, data types, or objects
- 312 • operation error precedence

### 313 2 Normative References

314 The following referenced documents are indispensable for the application of this document. For dated  
315 references, only the edition cited applies.

316 DMTF, [Common Information Model \(CIM\) Schema, version 2.12](#), April 20, 2006

317 DMTF, [DSP0215](#), *Server Management Managed Element (SM ME) Addressing Specification v1.0*,  
318 November 10, 2006

319 DMTF, [DSP0224](#), *Server Management Command Line Protocol (SM CLP) Command Response XML*  
320 *Schema, v1.0*, 2006

321 DMTF, [DSP1005](#), *Command Line Protocol (CLP) Service Profile, v1.0*, October 10, 2006

322 IETF, [RFC2234](#), *Augmented BNF for Syntax Specifications: ABNF*, November 1997

323 IETF, [RFC2396](#), *Uniform Resource Identifiers (URI): Generic Syntax*, August 1998

324 [ISO 639-2](#), *Codes for the Representation of Names of Languages Part 2: Alpha-3 Code*, March 2002

## Server Management Command Line Protocol (SM CLP) Specification

325 ISO, [ISO/IEC Directives, Part 2: Rules for the structure and drafting of International Standards](#), Fifth  
326 edition, 2004

### 327 **3 Terms and Definitions**

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

#### 329 **3.1**

##### 330 **Absolute Target Address**

331 a designation of target address that begins at the root of the containment hierarchy

#### 332 **3.2**

##### 333 **Addressing Associations**

334 an association instance that is used by the MAP to construct the UFiP to an instance referenced by the  
335 association instance

#### 336 **3.3**

##### 337 **Admin Domain**

338 the set of Managed Elements, Logical Devices, and Services for which the MAP has management  
339 responsibilities

#### 340 **3.4**

##### 341 **Association**

342 a relationship between two Managed Elements, which is itself a manageable entity within the Admin  
343 Domain

#### 344 **3.5**

##### 345 **Association Class**

346 identifies the CIM Class of an Association

#### 347 **3.6**

##### 348 **Client**

349 any system that acts in the role of a client to a MAP

#### 350 **3.7**

##### 351 **CLP Service**

352 the logical entity within a MAP that implements the CLP

#### 353 **3.8**

##### 354 **CLP Target**

355 a Managed Element or Association whose properties, behavior, UFiT, and so on are wholly defined by  
356 the profiles approved for use with the CLP

#### 357 **3.9**

##### 358 **Command Line Protocol**

##### 359 **CLP**

360 the human-oriented command line protocol defined by the System Management Architecture for Server  
361 Hardware, used for managing systems

#### 362 **3.10**

##### 363 **Command or Command Line**

364 a text message containing the complete expression of a management action, including a command verb,  
365 an optional command target, options and option arguments, and properties and values

- 366 **3.11**  
367 **Command Processor**  
368 the logical entity within a MAP responsible for parsing, interpreting, and executing incoming commands  
369 and returning responses
- 370 **3.12**  
371 **Command Response**  
372 response returned by the CLP Service to a Client when a Command is issued  
373 The Command Response consists of Command Status and Command Results.
- 374 **3.13**  
375 **Command Results**  
376 the actual results of a successful command returned as part of the Command Response
- 377 **3.14**  
378 **Command Status**  
379 information returned by the CLP Service to a Client describing the overall status of a Command
- 380 **3.15**  
381 **Command Status Data**  
382 detailed information returned by the CLP Service to a Client describing the status of the Command as  
383 part of the Command Response
- 384 **3.16**  
385 **Core Properties**  
386 properties for which the profile that owns the definition of the class does not stipulate any behavioral  
387 requirements  
388 However, the properties are defined in the MOF.
- 389 **3.17**  
390 **Current Default Target**  
391 **CDT**  
392 the CLP session environment setting that establishes a default base address for all command targets that  
393 are expressed as a Relative Target Address and is used as the command target if a target term is not  
394 specified in a command entered
- 395 **3.18**  
396 **Implicit Command Target**  
397 target acted upon that is inherent to the command being executed  
398 The command does not act upon either the Current Default Target or a target specified as part of the  
399 command. The `cd` command is an example of a command that acts upon an Implicit Command Target.
- 400 **3.19**  
401 **Keyword**  
402 a text-string token that is recognized and reserved by the CLP to have a specified meaning when used in  
403 command output and input
- 404 **3.20**  
405 **Local Addressing Service**  
406 the entity responsible for discovering, enumerating, and determining the addresses of Managed Elements  
407 and Associations within the Admin Domain

## Server Management Command Line Protocol (SM CLP) Specification

- 408 **3.21**  
409 **Manageability Access Point**  
410 **MAP**  
411 a service of a system that provides management in accordance with specifications of the DMTF System  
412 Management Architecture for Server Hardware
- 413 **3.22**  
414 **Managed Element**  
415 **ME**  
416 the finest granularity of addressing, which can be the target of commands or messages, or a collection  
417 thereof
- 418 **3.23**  
419 **Managed Element Access Method**  
420 the method by which a Managed Element performs a unit of work
- 421 **3.24**  
422 **Managed System**  
423 a collection of Managed Elements that compose a computer system for which the MAP has management  
424 responsibilities
- 425 **3.25**  
426 **Management Service Core**  
427 the logical entity that contains the core services set of the MAP
- 428 **3.26**  
429 **Non-addressing Association**  
430 an association instance that is not used by the MAP in constructing the UFiP to any instances referenced  
431 by the association instance
- 432 **3.27**  
433 **OEM Properties**  
434 properties added to instances of a class by an OEM vendor
- 435 **3.28**  
436 **OEM Target**  
437 a Managed Element or Association whose properties, behavior, UFcT, and so on are outside the scope of  
438 this specification and are vendor dependent
- 439 **3.29**  
440 **OEM Verbs**  
441 verbs defined by an OEM vendor that are outside the scope of this specification
- 442 **3.30**  
443 **Operation**  
444 an identifiable activity of a MAP
- 445 **3.31**  
446 **Option**  
447 a term of the Command Line that selects a particular behavior of a command verb
- 448 **3.32**  
449 **Option Argument**  
450 input data passed to the command verb in relation to an option that selects a particular value for that  
451 option

- 452 **3.33**  
453 **Property**  
454 an attribute of the command target
- 455 **3.34**  
456 **Relative Target Address**  
457 a designation of target address in relation to the Current Default Target as opposed to an Absolute Target  
458 Address
- 459 **3.35**  
460 **Required Properties**  
461 properties for which the profile that owns the definition of the class requires that instances of the class  
462 have values
- 463 **3.36**  
464 **Reserved String**  
465 a text-string token that is recognized and reserved by the CLP to have a specified meaning when used as  
466 an option argument, argument value, or property value
- 467 **3.37**  
468 **Reserved Target**  
469 a valid value for a target term that has a special meaning defined by this specification  
470 The meaning of a Reserved Target cannot be changed by a user nor can additional Reserved Targets be  
471 created.
- 472 **3.38**  
473 **Resultant Address**  
474 the Target Address after applying the target address precedence rules
- 475 **3.39**  
476 **Resultant Target**  
477 the effective target for a command after applying the target address precedence rules
- 478 **3.40**  
479 **Service Access Point**  
480 **SAP**  
481 the representation of the endpoint or interface into a service, such as the CLP
- 482 **3.41**  
483 **SM CLP Verb**  
484 a verb defined by this specification
- 485 **3.42**  
486 **Target**  
487 the Managed Element or Association upon which a command acts
- 488 **3.43**  
489 **Target Address**  
490 a string value used as the target term in a Command Line to identify the target for a command
- 491 **3.44**  
492 **Target Class Addressing**  
493 a method of selecting Managed Elements within a container based on the UFcT of the element

## Server Management Command Line Protocol (SM CLP) Specification

- 494 **3.45**  
495 **Text Session or Text-Based Session**  
496 an active connection to a service whereby the user can enter text-based messages and receive text-  
497 based data  
498 Examples of commonly used text sessions are Telnet and Secure Shell.
- 499 **3.46**  
500 **Transport**  
501 the layers of the communication stack responsible for reliable transportation of commands and messages  
502 between the Client and the MAP
- 503 **3.47**  
504 **User-Friendly instance Tag**  
505 **UFIT**  
506 user-friendly identifier for a specific instance of a CIM class  
507 A User-Friendly instance Tag is constructed by concatenating an integer suffix to the UFcT for the CIM  
508 class.
- 509 **3.48**  
510 **User-Friendly selection Tag**  
511 **UFsT**  
512 short-hand notation for selecting all instances of a given class  
513 A User-Friendly selection Tag is constructed by concatenating the UFcT for a class with the character \*.
- 514 **3.49**  
515 **User-Friendly class Tag**  
516 **UFcT**  
517 a short, user-friendly alias for a CIM class name  
518 It has the same properties and methods as the CIM class it represents.
- 519 **3.50**  
520 **User-Friendly instance Path**  
521 **UFiP**  
522 the unique path to an instance formed by concatenating the UFITs of each instance from the root instance  
523 to the terminating instance
- 524 **3.51**  
525 **Verb**  
526 the string name of a command, used as the first term of a Command Line

## 527 **4 Symbols and Abbreviated Terms**

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

- 529 **4.1**  
530 **CDT**  
531 Current Default Target
- 532 **4.2**  
533 **CIM**  
534 Common Information Model



## Server Management Command Line Protocol (SM CLP) Specification

535	<b>4.3</b>
536	<b>CLP</b>
537	Command Line Protocol
538	<b>4.4</b>
539	<b>MAP</b>
540	Manageability Access Point
541	<b>4.5</b>
542	<b>ME</b>
543	Managed Element
544	<b>4.6</b>
545	<b>NIC</b>
546	Network Interface Card
547	<b>4.7</b>
548	<b>NVT</b>
549	Network Virtual Terminal
550	<b>4.8</b>
551	<b>OEM</b>
552	Original Equipment Manufacturer
553	<b>4.9</b>
554	<b>SMASH</b>
555	System Management Architecture for Server Hardware
556	<b>4.10</b>
557	<b>SSHv2</b>
558	Secure Shell Version 2
559	<b>4.11</b>
560	<b>UFcT</b>
561	User-Friendly class Tag
562	<b>4.12</b>
563	<b>UFIT</b>
564	User-Friendly instance Tag
565	<b>4.13</b>
566	<b>UFiP</b>
567	User-Friendly instance Path
568	<b>4.14</b>
569	<b>UFsT</b>
570	User-Friendly selection Tag
571	<b>4.15</b>
572	<b>WBEM</b>
573	Web Based Enterprise Management

## 574 **5 Server Management Command Line Protocol (SM CLP)** 575 **Specification**

576 The following clauses detail the requirements for the SM CLP.

### 577 **5.1 Semantics**

578 The Command Line Protocol (CLP) defines the form and content of messages transmitted from and  
579 responses received by a Client within the context of a text-based session between that Client and the  
580 CLP Service for a Manageability Access Point (MAP).

581 The CLP consists of a set of command verbs that manipulate command targets representing Managed  
582 Elements (ME) that are within the scope of access by a MAP.

583 Each CLP interaction consists of a Command Line transmitted to the CLP Service and a subsequent  
584 response transmitted back to the Client. Each command transmitted generates one and only one  
585 response data transmission to the Client.

#### 586 **5.1.1 Command Verb**

587 A CLP command verb retrieves information about a target or initiates a state change of the target. A CLP  
588 interaction shall consist of one and only one command verb.

#### 589 **5.1.2 Command Options**

590 CLP command options control the behavior of the command verb. All CLP option names are standard  
591 across the CLP command verb set. Implementations of the CLP shall not redefine the usage of a CLP  
592 option name across different CLP command verbs.

#### 593 **5.1.3 Command Target**

594 This clause details requirements related to the usage and interpretation of a command target.

##### 595 **5.1.3.1 General**

596 The command target identifies the specific Managed Element or Association that is to be affected by the  
597 command verb. All CLP commands have a command target, whether explicitly or implicitly identified. An  
598 explicitly identified target is a target address path that is included in the Command Line entered. An  
599 implicitly identified target is a target that is not identified in the Command Line entered but either is  
600 dictated by the command verb itself or is referenced from the session environment variable "Current  
601 Default Target". Implementations shall interpret command verbs submitted to the CLP only for the  
602 Resultant Target. The CLP also defines Reserved Targets. Reserved Targets are strings whose  
603 interpretation is defined by this specification. Reserved Targets can be used to construct the command  
604 target term. Implementations shall not define Reserved Targets beyond the ones defined in this  
605 specification. Implementations shall interpret Reserved Targets in accordance with the meaning assigned  
606 to them by this specification.

607 This version of the SM CLP Specification supports the *Server Management Managed Element (SM ME)*  
608 *Addressing Specification v1.0* (DSP0215).

##### 609 **5.1.3.2 Current Default Target**

610 A Current Default Target address shall always be in effect during a CLP session. This target is used by  
611 the Command Processor to determine the Resultant Target for the command according to the rules of  
612 target address precedence defined in 5.1.3.3.

## Server Management Command Line Protocol (SM CLP) Specification

613 A session's Current Default Target is only modified if set explicitly by a user. The rules for establishing  
614 and maintaining a session's Current Default Target are as follows:

- 615 • Implementations shall set the Current Default Target to the instance address of the root of the  
616 address space of the MAP upon CLP session activation. Implementations shall use the same  
617 initial value for the CDT for all users and shall not allow the initial value to be configurable by a  
618 user.
  - 619 – Therefore, Current Default Target is never <null> at CLP session start.
  - 620 – If a user sets the CDT to another target address and then ends the session, on the next  
621 login by the user to a CLP session of the same MAP, the implementation shall set the CDT  
622 to the UFiP of the Managed Element that represents the root of address space of the MAP.
- 623 • The Command Processor shall not allow the user to explicitly set the Current Default Target to  
624 an invalid target address during the session.
- 625 • If the user attempts to set the Current Default Target to a target address for a Managed Element  
626 that is not responding or is not recognized as being in the scope of the MAP, then the attempt  
627 fails, and the implementation shall not change the Current Default Target and shall return a  
628 Command Status of COMMAND EXECUTION FAILED and a CIM Status of  
629 CIM\_ERR\_NOT\_FOUND.
- 630 • If a Managed Element becomes unresponsive at some point after it has been set as the Current  
631 Default Target, then the implementation shall return an appropriate error code and shall keep  
632 the Current Default Target address set to its current value until the user explicitly changes it to a  
633 different, valid target address. This prevents spurious drops in communication with the Current  
634 Default Target from causing an automatic change in the Current Default Target. Because  
635 unpredictable, undesired results would occur if the Current Default Target is automatically  
636 changed, the Command Processor shall not automatically change the value of the Current  
637 Default Target, for any reason.

638 **EXAMPLE** A user sets the CDT to "/system1/disk3". Some time later, /system1/disk3 becomes  
639 unresponsive. As long as the user does not target the CDT with a command, there is no impact on the user's  
640 current session. If the user decides to target /system1/disk3 by omitting the target term of the current command,  
641 the CLP implementation would discover that the target ME, /system1/disk3, is unresponsive and return an error  
642 code.

### 643 5.1.3.3 Target Address Precedence

644 The implementation shall determine the Resultant Target of a command in the order that follows:

- 645 1) If the command verb has an Implicit Command Target, then the Implicit Command Target shall  
646 be selected as the Resultant Target.
- 647 2) If a command target term is specified in the Command Line, the implementation shall apply the  
648 Target Address Evaluation Rules to derive the Resultant Target. These rules are detailed in  
649 5.2.1.3.6.
- 650 3) If the Command Line did not include a command target term, the implementation shall select  
651 the CDT as the Resultant Target.

652 If the Resultant Target is determined by the implementation to be invalid, then the implementation shall  
653 not execute the command and shall return a Command Status of COMMAND EXECUTION FAILED and  
654 a CIM Status of CIM\_ERR\_NOT\_FOUND in the Command Response data. CLP commands that have an  
655 Implicit Command Target may still accept a command target term (for example, the `cd` command) or may  
656 not accept a command target term (for example, the `exit` command). Each command's use of the  
657 command target term is documented in the subclause of Clause 6 devoted to the command.

## Server Management Command Line Protocol (SM CLP) Specification

### 5.1.3.4 Target Managed Element Object Model and Semantics

659 The CLP is designed for administrators and scripts that manage systems. At the same time, the CLP  
660 conforms to the object model described by the *4HCommon Information Model (CIM) Schema, version*  
661 *2.12*.

662 The CLP defines a set of general command verbs used to manipulate Managed Elements. In many  
663 cases, CLP verbs relate directly to typical object interactions, such as "set property value", "read property  
664 value", "put into a particular state", and so on. In other cases, CLP verbs are interpreted in the context of  
665 the Managed Element and map to particular methods of that Managed Element's class.

666 The CLP verb definitions in Clause 6 describe each CLP command verb in detail.

667 DSP0216 describes the full mapping of the CLP to the CIM. For each CIM class, DSP0216 describes the  
668 behavior of commands applied to a target instance of the class. The specification also describes the  
669 property names of those targets that are referenced or manipulated by the command.

670 In the CLP, Managed Elements have the following aspects:

- 671 • Properties

672 These are properties of the Managed Element itself and are described in more detail in 5.1.4.

- 673 • Contained Targets

674 This is the set of Managed Elements immediately contained in the Managed Element according  
675 to the rules of instance containment described in 5.1.3.5.

- 676 • Associations

677 This is the set of associations that reference the Managed Element. They are described in more  
678 detail in 5.1.5.

- 679 • Verbs

680 This is the set of commands that are applicable to the Managed Element. The SM CLP verbs  
681 are described in Clause 6.

### 5.1.3.5 Target Addressing

683 CLP target addressing is defined by DSP0215. CLP implementations shall operate only on command  
684 target terms that adhere to DSP0215 or to the rules for identifying OEM targets described in 5.2.6.

685 The specific arrangements of Managed Elements that a MAP may expose are documented in DSP0215  
686 and SMASH Implementation Requirements (DSP0217). The SM CLP separates Managed Elements into  
687 two categories of targets: CLP Targets and OEM Targets. CLP Targets are Managed Elements whose  
688 properties, behavior, UFcT, and so on are wholly defined by the profiles approved for use with the CLP.  
689 OEM Targets are Managed Elements whose properties, behavior, UFcT, and so on are outside the scope  
690 of the profiles approved for use with the CLP and are vendor dependent.

### 5.1.3.6 Aggregated Targets

692 Command targets may be an aggregation of underlying components. These underlying components may  
693 be visible in the address space of the MAP. When the command target is composed of aggregated parts,  
694 the Command Processor shall interpret the command for the aggregated target as a single job and return  
695 a Command Response accordingly.

696 The implementation may rely on the target Managed Element to implement the aggregated command  
697 function. One example of an aggregated target is an operating system. When a user issues a `stop` to an  
698 operating system instance, a single job is spawned. The operating system may attempt to shut down  
699 applications running within it. This action taken by the operating system is not modeled with jobs, and the  
700 results for individual applications are not displayed in the Command Results.

### 701 5.1.3.7 Target Grouping by Class

702 "Grouping" describes the ability of a user to explicitly select more than one target for a command at the  
703 time the command is issued. The only method defined by the CLP for addressing multiple targets with a  
704 single command is Target Class Addressing.

705 If the final term of the command target term is a UFST, in general the Command Processor interprets the  
706 command target term as a selector for all Managed Elements of the class specified that are in the  
707 immediate container.

708 Implementations shall support Target Class Addressing for the `show` command. Implementations may  
709 support Target Class Addressing for the `create` and `delete` commands. Implementations shall not  
710 support Target Class Addressing for CLP commands other than the `show`, `create`, and `delete`  
711 commands. When Target Class Addressing is utilized for a command, the implementation shall select the  
712 instances to be the target of the command by using the Rules for Selecting Instances by UFCT defined in  
713 DSP0215, where the Selection UFCT is the UFCT identified by the UFST specified in the command target  
714 term.

715       EXAMPLE     The command target term `"/system3/disk*` instructs the Command Processor to issue the  
716       command for all targets with an UFCT of `"disk"` in the container `"/system3"`.

### 717 5.1.4 Command Target Properties

718 Target properties are identifying and descriptive information related to and defined by the target. Target  
719 properties are identified by property names. Each class of target defines a set of valid property names.  
720 Valid property names are found in the CIM Schema Managed Object Files (MOFs). Vendors may support  
721 vendor-specific property names according to the rules defined in 5.2.6 of this specification. The SM CLP  
722 recognizes three categories of properties:

- 723       • Required Properties are properties that the profile defining the class of the target deems  
724       required for compliance with the profile. These properties will be present for the instance across  
725       implementations.
- 726       • Core Properties are properties that are defined for the class of the target in the CIM schema.  
727       The profile that defines the class does not require these properties; however, they may be used  
728       because they are defined in the MOF. Note that this includes any deprecated properties which  
729       are still defined in the MOF. They may be present across implementations.
- 730       • OEM Properties are properties defined by an OEM vendor for a target. These properties will not  
731       be consistent across different vendors' implementations.

### 732 5.1.5 Associations

733 DSP0215 specifies the Association Classes that may be used to construct paths to address any Managed  
734 Element appearing within the scope of the MAP. DSP0215 identifies these as Addressing Associations.  
735 Additional associations that are not used for addressing may exist and express relationships between  
736 Managed Elements. DSP0215 identifies these as Non-addressing Associations. Associations represent a  
737 special type of target. Association instances are not assigned UFITs. For a given association class,  
738 instances are uniquely identified by the Managed Element instances they reference. They can be  
739 addressed using an extension of the target addressing syntax. 5.2.1.3.5 describes how to use the  
740 association separator `"=>"` to address association instances. 6.10 and 6.11 illustrate the use of the `set`  
741 and `show` commands, respectively. Associations have properties which follow the rules for Command  
742 Target Properties as identified in 5.1.4.

### 743 5.1.6 Command Processing

744 This clause states the requirements for the processing of a CLP Command Line.

## Server Management Command Line Protocol (SM CLP) Specification

### 745 5.1.6.1 General

746 Implementations of the CLP shall return a Command Status of COMMAND PROCESSING FAILED and a  
747 Processing Error of COMMAND SYNTAX ERROR if a completely formed command is not contained in a  
748 single text message transmission of the underlying transport protocol. Implementations shall validate  
749 every Command Line against the `clp-command-line` production of the grammar specified in  
750 Annex A. When a Command Line does not comply with the `clp-command-line` production of the  
751 grammar defined in Annex A, the implementation shall return a Command Status of COMMAND  
752 PROCESSING ERROR and a Processing Error of COMMAND SYNTAX ERROR. There are error  
753 conditions identified by this specification that can be detected by validation against the grammar for which  
754 this specification identifies specific values for Processing Error that are required to be returned instead.  
755 When specified, the requirement to return specific values supersedes the requirement to return the  
756 general COMMAND SYNTAX ERROR.

757       EXAMPLE     The `create` command requires a command target term to be specified. The specific Processing  
758 Error of MISSING REQUIRED TARGET is required to be returned if the command target term is not included.  
759 The grammar also requires that a command target term be specified in the `create-cmd` production. Thus, if a  
760 command target term is not specified, the Command Line will fail validation against the grammar and a  
761 COMMAND SYNTAX ERROR would be appropriate. However, this general Processing Error is superseded by  
762 the specific Processing Error specified for the error condition.

### 763 5.1.6.2 Job Visibility

764 A command is processed by the Command Processor component of the SM architecture. The Command  
765 Processor returns a response and control to the Client for each command received. Commands and the  
766 subordinate activities generated when processing commands are tracked by the implementation. A job is  
767 defined as an identifiable activity of a MAP. The implementation spawns and manages jobs for a  
768 command and any subsequent actions that are taken to carry out the command request. When an  
769 implementation receives a command, the command becomes a job. The command job may generate  
770 additional subordinate Managed Element jobs in order to complete the task requested by the command  
771 verb, but the implementation shall not expose these jobs through the CLP. The command job shall  
772 continue to exist until any and all jobs spawned by the command have completed, a Command Response  
773 has been returned to the Client, and the Time Before Removal has not expired.

774 By default, the implementation shall return a Command Response and session control to the Client within  
775 a reasonable period of time, regardless of the status of command execution. Returning a response and  
776 control prevents the CLP session from blocking indefinitely if a command takes an unreasonable amount  
777 of time. The definition of "a reasonable period of time" is implementation specific. Any mechanisms for  
778 modifying this value are outside the scope of this specification and are implementation specific.

779 When the implementation returns a Command Response synchronous with completion of the command  
780 job, the Command Response shall contain the Command Status and the complete Command Results.

781 When the implementation returns a Command Response before the command job completes, the  
782 Command Response shall contain the Command Status of COMMAND SPAWNED and the Job Identifier  
783 for the continuing command job.

784 The implementation shall manage the command job until it completes and persist the Command Status  
785 when complete, identified by the Job Identifier. The implementation is not required to maintain the  
786 Command Results for the command. Implementations shall recognize the Job Identifier as an identifier  
787 used to obtain status information about the continuing command and to retrieve the Command Status  
788 when the command job is complete. The Job Identifier shall be the Instance Suffix for the UFI of the  
789 instance of `CIM_ConcreteJob` used to represent the job in the job queue. After the Command Status  
790 holding time has expired, the corresponding job is deleted and the Job Identifier is released.  
791 Implementations shall also implement a user-controlled holding time for Command Status, controlled by a  
792 per-command option. Use of this option is documented in **Error! Reference source not found.** Every  
793 command job has a Time Before Removal associated with it. The Time Before Removal indicates the  
794 amount of time that a command job is managed by the MAP after completion. The command job itself is

## Server Management Command Line Protocol (SM CLP) Specification

795 represented with an instance of CIM\_ConcreteJob. The Time Before Removal of the command job is  
796 modeled with the TimeBeforeRemoval property of the CIM\_ConcreteJob instance.

797 Jobs are themselves Managed Elements of the system; therefore, the implementation shall support  
798 display of information about a job and the ability to request a job to stop before completion using CLP  
799 commands. If the implementation is unable to start a job to execute a command, the implementation shall  
800 return a Command Status of COMMAND PROCESSING FAILED and a Processing Error of QUEUE  
801 FULL.

### 802 5.1.6.3 Error Handling

803 The CLP Service checks CLP commands for syntax and semantic errors. When a command is formed  
804 incorrectly or the command cannot be executed for the specified target because the target is not in an  
805 appropriate state, the implementation shall return an error/exception status in the Command Status data  
806 and no Command Results.

807 When a command is syntactically correct and semantically appropriate, the implementation shall attempt  
808 to perform the appropriate operations for the command target. If one or more operations fail, the  
809 implementation shall return a Command Response containing both a Command Status and Command  
810 Results, including any error/exception information generated by the command target.

811 Implementations shall include all detected syntax errors first in the Command Status data.  
812 Implementations shall include all detected semantic errors in the Command Status data.

813 If the Command Processor detects a syntax error, the implementation shall report an error and the  
814 implementation shall not alter the state of the Target. If the Command Processor detects a semantic error,  
815 the implementation shall report an error and should not alter the state of the Target.

816 Command Status output is defined by the CLP and is documented in 5.2.2.

817 Command Results output is defined for each CLP command and is documented in Clause 6.

### 818 5.1.7 SESSION Reserved Target

819 Sessions with the CLP Service are represented as Managed Elements within the address space of the  
820 MAP. This enables users of the CLP to manage attributes of the session using standard CLP commands.  
821 Users will frequently wish to manage attributes of their own session with the CLP Service. To simplify  
822 accessing the Managed Element that represents the user's session, the CLP defines a reserved keyword  
823 "SESSION". Implementations shall interpret the keyword "SESSION" as the fully qualified path to the  
824 Managed Element that represents the session of the user issuing the command.

### 825 5.1.8 UFiT Assignment

826 Individual Managed Elements within the address space of the map are identified by a UFiT. The UFiT is  
827 constructed by concatenating an integer suffix to the UFcT for the class of the Managed Element. The  
828 rules for assigning and maintaining UFiTs are defined in DSP0215.

### 829 5.1.9 Input Data

830 This clause states requirements for the handling of input data.

#### 831 5.1.9.1 General

832 Implementations of the CLP shall not allow inclusion of input data embedded in the text session  
833 (sometimes referred to as "streaming"). A data file or stream can be selected for input to a CLP command  
834 by reference only using a command-specific option.

835       EXAMPLE     To input a firmware image for reloading firmware, the option `-source <URI>` is used.

## Server Management Command Line Protocol (SM CLP) Specification

### 836 5.1.9.2 Data Passed in on Command Line

837 Data may be provided as input to a command through an option argument, option argument value, or a  
838 property value. Each command verb defines the options and option arguments that it accepts. The class  
839 of command target defines the properties that are accepted by the command. Implementations shall  
840 enforce a maximum length of 255 characters for any single term in a command. If a single term exceeds a  
841 maximum length of 255 characters, the implementation shall not execute the command and shall return a  
842 Command Status of COMMAND PROCESSING FAILED and a Processing Error of COMMAND SYNTAX  
843 ERROR.

### 844 5.1.10 Output Data

845 This clause states requirements related to output data.

#### 846 5.1.10.1 General

847 The semantics of CLP command output data are defined by the CLP output data schema. The CLP  
848 output data schema defines the attributes and organization of the data elements returned in response to a  
849 CLP command. When a command is issued, in the absence of transport errors, the implementation shall  
850 return a Command Response data element as the response before any other text that is appended.  
851 Implementations shall include a Command Status data element in the Command Response data element  
852 and may include a Command Results data element. Implementations shall always return Command  
853 Status data first in a Command Response.

854 The CLP syntax defines the keywords and value specifications (data types and ranges or domains) that  
855 are used to document CLP command output.

856 CLP output data is rendered in character form according to the rules set forth in 5.2.4 and 5.2.1.1. By  
857 default, a CLP session renders all output data in an output format called "text". Text output format is  
858 defined to be human-readable text that is not suitable for machine-parsing and will vary across  
859 implementations. In order to parse output data according to the CLP output data schema, selection of one  
860 of the CLP's structured output formats is recommended. These formats are "keyword" and "clpxml".  
861 Implementations shall use the attribute keywords consistently to identify output data elements in each of  
862 the output formats. For example, the keyword "status" is rendered as a keyword in a "keyword=value"  
863 expression in "keyword" format and as an XML tag in "clpxml" format.

864 Text mode output is optimized for human readability and is likely to vary from implementation to  
865 implementation and in situation to situation. For example, when in text output mode, Command Status  
866 Data may not display the numeric value of the status code when the command is successful but instead  
867 simply describe the resulting condition of the target after the command has completed. For example,  
868 when the command "stop system1" completes successfully, the text mode output may simply state  
869 "system1 stopped". By contrast, when a structured output mode is selected, the implementation shall  
870 include all of the required and supported optional elements of the Command Status Data in the output.

871 The CLP Output Data Schema is documented in 5.2.2.

872 The following clauses contain data definitions for the output data elements in the CLP Output Data  
873 Schema.

#### 874 5.1.10.2 Command Status Data Elements

875 This clause states requirements related to Command Status Data Elements.

##### 876 5.1.10.2.1 General

877 Command Status data elements communicate the status of the command to the Client. Depending on the  
878 command verb and user-selected options, a command may continue to run after returning a response  
879 and control to the Client. In this case, the Command Status data also includes a Job Identifier that can be  
880 used to display the status of the command job at a later time.



## Server Management Command Line Protocol (SM CLP) Specification

881 The implementation shall return the Command Status data element as the first data element in a  
882 Command Response. Implementations shall not include any data elements or properties in the Command  
883 Status data element other than those defined here.

884 The Command Status data element includes

- 885 • a Status property
- 886 • a Status Tag property
- 887 • a Processing Error property
- 888 • a Processing Error Tag property
- 889 • one or more Message data elements
- 890 • a Job data element

891 The Status property describes the outcome of the command. The Command Status documents the  
892 disposition of the command from the perspective of the CLP Session protocol. Status for each command  
893 is one of four values: COMMAND COMPLETED, COMMAND SPAWNED, COMMAND PROCESSING  
894 FAILED, or COMMAND EXECUTION FAILED. If the command fails, then the Client can examine the  
895 subsequent Command Status data element to determine the cause of the failure. The Status Tag  
896 property is the descriptive string corresponding to the numeric value of the Status property. The complete  
897 list of Status and Status Tag values are listed in Table 4.

898 Processing Errors are conditions detected by the Command Processor prior to creating a job to execute  
899 the command. These are generally syntax-related errors. Two properties are defined for a Processing  
900 Error. The Processing Error property identifies the specific error that occurred when processing the  
901 command. The Processing Error Tag property is the descriptive string corresponding to the numeric value  
902 of the Processing Error property. The complete list of Processing Errors is in Table 6.

903 Implementations may support the Status Tag and Processing Error Tag properties.

### 904 5.1.10.2.2 Message Data Elements

905 The Message element is a text message that describes the disposition of the command. By default, the  
906 status message is presented in English text.

907 The Message data element includes

- 908 • a Message property
- 909 • an Owning Entity property
- 910 • a Message Id property
- 911 • one or more Message Argument properties

912 When the Message data element is included in a Command Response, the implementation shall include  
913 the Message, Owning Entity, and Message Id properties, and may include one or more Message  
914 Argument properties. Implementations shall ensure that the value of the Message Id property together  
915 with the value of the Owning Entity property is unique. The Client can use these values to identify  
916 corresponding language translations of the Command Status message.

917 To guarantee uniqueness, the Owning Entity property shall include a prefix string that uniquely identifies  
918 the entity that owns and defines the Owning Entity value. The prefix string shall include a copyrighted,  
919 trademarked, or otherwise unique name that is owned by the business entity or standards body defining  
920 the owning entity string. The implementation shall not return a status message that exceeds 256  
921 characters. Implementations may support the Message data element.

## Server Management Command Line Protocol (SM CLP) Specification

### 922 5.1.10.2.3 Job Data Elements

923 The Job data element describes the job created to execute the command.

924 The Job data element includes

- 925 • a Job Identifier property
- 926 • a Job Error data element

927 The Job Identifier is used to identify the MAP job that represents the command execution. The Job  
928 Identifier is used to query for Command Status at a later time. The value of the Job Identifier property is  
929 the Job Identifier for the job spawned by the MAP to process a command. The implementation shall  
930 include the Job Identifier property whenever it returns a Job data element in a Command Response.

931 Job Errors are conditions that are detected by the implementation or by the Managed Element target of  
932 the command. These errors are detected after a job is created to execute the command. The Job Error  
933 properties provide details of the cause of the command failure. The Job Error data element includes

- 934 • an Execution Error property
- 935 • an Execution Error Tag property
- 936 • one or more Message data elements
- 937 • a CIM Status property
- 938 • a CIM Status Description property
- 939 • a Severity property
- 940 • a Severity Description property
- 941 • a Probable Cause property
- 942 • a Probable Cause Description property
- 943 • one or more Recommended Action properties
- 944 • an Error Source property
- 945 • an Error Source Form property
- 946 • an Error Source Form Description property

947 When the Job Error data element is included in a Command Response, the implementation shall include  
948 the Execution Error, CIM Status, and Severity properties. When the Job Error data element is included in  
949 a Command Response, the implementation may include Message data elements and may include any  
950 other properties of the Job Error data element not explicitly required. A complete description of Job Error  
951 properties and values is found in 5.2.3.2.

952 The implementation shall include the Status property in the Command Status data element.  
953 Implementations may include the Status Tag property and may include the Message data element. The  
954 implementation shall not include the Processing Error or Processing Error Tag properties in the  
955 Command Status data element unless the Command Status is COMMAND PROCESSING FAILED.  
956 When the Command Status is COMMAND PROCESSING FAILED, the implementation shall include the  
957 Processing Error property in the Command Status data element.

958 When the Command Status is COMMAND EXECUTION FAILED, the implementation shall include the  
959 Job data element in the Command Status data element and shall include the Job Error data element in  
960 the Job data element.

961 When the Command Status is COMMAND PROCESSING FAILED, the implementation shall not include  
962 the Job data element in the Command Status data element.

## Server Management Command Line Protocol (SM CLP) Specification

963 When the Command Status is COMMAND SPAWNED, the implementation shall include the Job data  
964 element in the Command Status data element and shall not include the Job Error data element in the Job  
965 data element.

966 When the Command Status is COMMAND COMPLETED, the implementation shall include the Job data  
967 element in the Command Status data element and shall not include the Job Error data element in the Job  
968 data element. When the Command Status is Completed, the implementation shall include the Command  
969 Results data element in the Command Response.

### 970 5.1.10.3 Command Results Data Elements

971 The output data elements for Command Results are defined by command verb-specific Command  
972 Results schema.

973 Command Results data elements include

- 974 • command-specific output data elements as defined by the command verb's specification
- 975 • target-specific data elements as defined by the SM Profiles

976 The specification for each CLP command verb documents the applicable Command Results schema for  
977 that command. See Clause 6 for command specifications and their corresponding Command Result  
978 schema.

979 In addition, CLP commands return output data that is relevant to the particular Managed Elements or  
980 Associations that were identified as the command target. Target-specific data elements are returned in  
981 the Command Results data.

982 See DSP0216 for the keywords and value specifications that are relevant to each class of the target  
983 Managed Element.

### 984 5.1.11 Session Prompt

985 CLP session output shall include a session prompt. The CLP defines a specific output format and  
986 character string that are used to delineate the session prompt.

987 CLP session output shall be of the following form:

988 `<OUTPUT><IMP PROMPT><CLP PROMPT>`

989 where

- 990 • **OUTPUT** is the Command Response. The Command Response shall be formatted  
991 according to one of the CLP-defined output formats.
- 992 • **IMP PROMPT** is the implementation's optional prompt string, which shall not contain the  
993 CLP PROMPT string.
- 994 • **CLP PROMPT** is the CLP standard prompt string, "-> ". The CLP prompt string shall  
995 appear after each Command Response.

996 Implementations shall include the CLP prompt string, "-> " (hyphen, greater than, space) after every  
997 Command Response returned to the Client. Implementations shall not return any text after the prompt.

998 Implementations may omit an implementation-provided prompt string. If the implementation provides an  
999 implementation prompt string, the implementation shall insert the prompt string after the CLP Command  
1000 Response data and before the CLP prompt string.

1001 Implementations may vary the implementation prompt string from Command Response to Command  
1002 Response. The implementation is not required to maintain a consistent prompt string.

## Server Management Command Line Protocol (SM CLP) Specification

1003 The CLP is a command-response text-based message protocol. An implementation of the CLP Service  
1004 shall return a response to each command presented by the Client. Implementations of the CLP Service  
1005 shall not accept any further commands from the Client until after the implementation returns a Command  
1006 Response for the currently outstanding command.

1007 Because the CLP is primarily for use by a human user, the CLP Service shall return a Command  
1008 Response within a "reasonable amount of time". The CLP Service shall be capable of spawning any  
1009 commands that it determines to be long running. When commands are spawned, the CLP Service shall  
1010 return an interim Command Response containing the Job Identifier that is to be used by the Client to  
1011 retrieve the Command Status and results when the command completes.

### 1012 5.1.12 Extending the Command Line Protocol

1013 The CLP can be extended by a vendor in one of the following ways:

- 1014 • supplying vendor-specific command verbs, options, target addresses, or properties
- 1015 • adding vendor-specific information to standard CLP command verb output

1016 Vendor extensions are conspicuously named as described in 5.2.6 so that the user is aware that the use  
1017 of the extension is non-standard. The syntax clause of this document defines areas of vendor extensibility  
1018 and the requirements in effect for those extensions.

## 1019 5.2 Syntax

1020 The CLP implements a small and easily remembered set of verbs (Clause 6) that apply across the entire  
1021 target address space (5.1.3). This allows users to quickly understand the function available to them and  
1022 then apply that knowledge across a wide variety of environments. These verbs provide a consistent set of  
1023 output (5.1.10), which further simplifies understanding by both the new and experienced user as they  
1024 move from implementation to implementation and from simple to complex behaviors. As users become  
1025 more experienced and sophisticated, they can further refine the behavior of these verbs using a set of  
1026 Command Line options that are also standard across the entire CLP verb space (Clause 7).

### 1027 5.2.1 Basic Command Syntax

1028 The SM CLP basic command syntax is described in the following clauses.

#### 1029 5.2.1.1 Character Set, Delimiters, Special, and Reserved Characters

1030 All implementations of the CLP shall interpret the characters provided by the transport as UTF8  
1031 representation of the characters, including those in Table 1, and shall interpret the characters in Table 1  
1032 according to the description included in Table 1.

1033 **Table 1 – CLP Reserved Characters and Character Sequences**

Character or Sequence	Name	Description / Uses
" "	space	Command line term separator.
`	escape character	Escape character (the backquote character), used in front of reserved characters to instruct the command parser to use the reserved character without special meaning. When the escape character is not followed by a reserved character, it is treated as a normal character in the string that contains it.
<cr> <lf> <cr><lf>	end-of-line	Each of these sequences is accepted as an end-of-line indicator.

## Server Management Command Line Protocol (SM CLP) Specification

Character or Sequence	Name	Description / Uses
<escape character><end-of-line>	line continuation	An escape character placed immediately before the end-of-line sequence indicates that the current line is continued to the following line. The following line will be appended to the current line.
,	comma	Delimits items in an option argument term that is to be interpreted as a list of option arguments. Also delimits values for an option argument.
=	assignment operator	A single equal sign '=' is used to separate a property name from a desired value for the property when used with verbs that modify or create an instance. It will not have a space before or after it in an expression of a property and its value.
==	equivalence operator	Two consecutive equal signs "==" without white space between them are used to separate a property name from a desired value when filtering instances for which results are returned.
-	hyphen	When preceded by a space, the hyphen is the CLP option indicator.
/ \	address term separator	Separates the UFiT terms of a target address.
=>	association separator	Used to separate an association from the portions of the command target term that identify the instances the association references.
.	dot	Recognized as a special target address token meaning "this container".
..	dot-dot	Recognized as a special target address token meaning "the container of this container".
( )	parentheses	In an option argument term that is a comma-separated list, delineates the values of an argument from the next option argument.
"	double quote	Delineates a string of text that may contain the CLP term separator (space) so that the CLP Command Processor will treat the delineated text as one string.
"-> "	CLP PROMPT (hyphen, greater-than, space)	Literal representation of the CLP prompt.
SESSION	SESSION	Reserved target representing the current session.

### 1034 5.2.1.2 Case Sensitivity

1035 The general CLP Command Line syntax is not case sensitive. Command verb, option, target, and  
 1036 property names may be expressed in any combination of uppercase and lowercase characters.  
 1037 Implementations shall accept command verb, option, option argument, target, and property names  
 1038 expressed in any combination of uppercase and lowercase characters.

1039       EXAMPLE    "show", "Show", and "SHOW" are all valid expressions of the CLP verb "show". "-o  
 1040       Format=clpxml", "-O format=clpxml", and "-OUTPUT FORMAT=clpxml" are all valid expressions of the output  
 1041       option and format argument.

1042 For readability, this specification documents all verb, option, target, and property names in lowercase.

1043 The CLP places no restrictions on case sensitivity for the interpretation of property values. Interpretation  
 1044 of property values (including case sensitivity) is determined by the profile that defines the target to which  
 1045 the property belongs. Requirements for the input format of common property types are specified in  
 1046 DSP0216.

## Server Management Command Line Protocol (SM CLP) Specification

### 1047 5.2.1.3 Command Line Terms

1048 This clause details the requirements for Command Line Terms.

#### 1049 5.2.1.3.1 General

1050 A CLP Command Line consists of four types of terms: verbs, options and option argument terms,  
1051 command target terms, and target property terms. Each term on the Command Line is separated from  
1052 other terms by the CLP command term separator character, " " (space). (See 5.2.1.1 for the list of CLP  
1053 special and reserved characters.) Implementations shall recognize the CLP command term separator  
1054 character, beginning of line, and end of line as delimiting terms on the Command Line.

1055 Implementations shall recognize the end-of-line character as terminating a single Command Line unless it  
1056 is preceded by the CLP escape character.

1057 A single Command Line may be continued across end-of-line by using the CLP escape character  
1058 immediately before the end-of-line character. The implementation shall not initiate command processing  
1059 until after the complete command has been received by the CLP. If an implementation receives a  
1060 Command Line that contains zero characters or consists entirely of the command term separator  
1061 character, implementations shall return a Command Response that contains exactly zero characters. The  
1062 effect of this requirement is such that a blank line is not reported as an error and instead results in a CLP  
1063 prompt being returned.

#### 1064 5.2.1.3.2 Verb

1065 The implementation shall expect the command verb to be the first term in a command. The  
1066 implementation will expect the command verb to be one of the specified CLP command verbs listed in  
1067 Clause 6 or an OEM command line extended form as defined in 5.2.6.3.3. Implementations shall not  
1068 support any verbs other than the CLP verbs defined in this specification and any command verbs  
1069 identified as OEM verbs according to 5.2.6.3.3. When the first term in a Command Line is not a CLP verb  
1070 and is not identified as an OEM verb according to the rules in 5.2.6.3.3, the implementation shall not  
1071 execute the command and shall return a Command Status of COMMAND PROCESSING FAILED and a  
1072 Processing Error of COMMAND NOT RECOGNIZED.

#### 1073 5.2.1.3.3 Options and Option Arguments

1074 Command options may be included immediately after the command verb. Command options are  
1075 recognized by the option indicator character, "-" (single hyphen). If options are specified in a command,  
1076 the options and their arguments shall occur immediately after the command verb and before the optional  
1077 command target term and target properties.

1078 Command options either require an argument or require no argument.

1079 Options that require no argument are separated from the subsequent options, command target term, or  
1080 target property names by the command term separator character. Options that require arguments are  
1081 separated from their option argument term by the command term separator character. An option  
1082 argument will be one or more argument names or argument/value pairs. The comma is used to delimit  
1083 arguments and argument/value pairs within the option argument term. The comma is also used to delimit  
1084 values within an argument value. Using the comma for two types of tokenization within the option  
1085 argument term could result in ambiguity when parsing an option argument term. To eliminate the potential  
1086 for ambiguity, parentheses are used to enclose argument values which can be comma-delimited lists.  
1087 When processing an option argument term, implementations will tokenize the option argument term using  
1088 the comma as a delimiter unless the comma is enclosed in parentheses, in which case the  
1089 implementation will ignore it. Implementations shall interpret a ", " (comma) as delimiting arguments in the  
1090 option argument term unless the comma is preceded by a left parenthesis "(", in which case the  
1091 implementation shall ignore any commas that occur prior to a matching right parenthesis ")". If while  
1092 parsing a Command Line an implementation encounters an option it does not recognize, the  
1093 implementation shall not execute the command and shall return a Command Status of COMMAND  
1094 PROCESSING FAILED and a Processing Error of INVALID OPTION.

### 1095 5.2.1.3.4 Target Address

1096 A Target Address is a string that follows the Target Address Syntax and is used to identify the target of a  
1097 command. The SM CLP supports several types of target addressing. It supports individual instance  
1098 addressing as defined in 5HDSP0215. Instance addressing is extended to include support for Relative  
1099 Target Addresses. It adds support for addressing instances of a particular class and support for  
1100 addressing an instance or instances of an association relative to target instances that the association  
1101 references.

1102 Implementations shall require that if the command target term is included in the Command Line, the  
1103 command target term is the first term that is not an option after the command verb. Implementations shall  
1104 require that when the command target term is included in the command, the command target term  
1105 appears before any target property names.

1106 The implementation shall observe the following ordered rules for determining if the first non-option or  
1107 option argument term after the verb is treated as a command target term:

- 1108 • If the first non-option or option argument term after the verb contains one of the following CLP  
1109 reserved addressing terms ( "." [dot], ".." [dot dot], address term separator, "=>" [association  
1110 separator], or SESSION) and the term is unescaped, the implementation shall interpret the term  
1111 as a command target term.
- 1112 • If the first non-option or option argument term after the verb ends in an integer or an "\*"   
1113 (asterisk) and does not contain an unescaped assignment operator or equivalence operator, the  
1114 implementation shall interpret the term as a command target term.
- 1115 • When the Command Processor discovers that the first non-option or option argument term after  
1116 the verb is not a command target term, the Command Processor shall assume that the term,  
1117 and any subsequent terms, are target property terms.

1118 These rules enable a Client wishing to ensure consistent results when specifying a command target term  
1119 to do so through the inclusion of one of the CLP reserved addressing terms. When the implementation  
1120 determines that a command target term has been specified, the implementation shall validate the  
1121 command target term against the all-legal-targets production of the grammar specified in Annex A. If the  
1122 command target term is invalid, the implementation shall return a Command Status of COMMAND  
1123 PROCESSING FAILED and a Processing Error of INVALID TARGET.

1124 The order of precedence for determining the Resultant Target of a command is defined in 5.1.3.

### 1125 5.2.1.3.5 Target Address Syntax

1126 Target addresses in the SM CLP are composed of the following parts:

- 1127 • UFiT—A UFcT concatenated with an integer suffix. Selects a specific Managed Element in a  
1128 given container.
- 1129 • UFsT—A UFcT concatenated with an asterisk. Selects all instances of the type specified by the  
1130 UFcT within a given container.
- 1131 • address term separator (/ or \)—When located at the beginning of a command target term,  
1132 represents the root of the address space. When used to separate two UFiTs in an address,  
1133 represents an Addressing Association between them.
- 1134 • association separator (=>)—Delimits an Association Class within a target address. The portion  
1135 of the target address preceding, and optionally following, the Association Class identifies one, or  
1136 both, Managed Elements referenced by the target Association instance.
- 1137 • dot (.)—Reserved term meaning "this target"
- 1138 • dot dot (..)—Reserved term meaning "the container of this target"

## Server Management Command Line Protocol (SM CLP) Specification

1139 Each UFiT is a short, text string identifier of a Managed Element in the address space of the MAP. A UFiT  
1140 is of the form "UFiT<integer suffix>" where the first component is a short, text string tag that  
1141 identifies the class of Managed Element and the second component is an integer suffix that uniquely  
1142 identifies the Managed Element in its container.

1143 An instance address is a sequence of Managed Element tags, or UFiTs, separated by the slash  
1144 character. Any UFiT that is followed by a slash character indicates that the remaining target address path  
1145 is contained within that Managed Element. A UFiP is an Absolute Target Address that references exactly  
1146 one Managed Element and does not contain any of the CLP addressing extensions (dot, dot dot, or  
1147 SESSION). The CLP Command Line grammar is formally defined in Annex A.

1148 The general syntax of an instance address is as follows (in ABNF form):

```
1149 [ <address term separator> ] *[ ( "." / ".." / <UFiT> ) <address term separator> ]  
1150 <UFiT>
```

1151 By successively interpreting each term of the command target term and performing any substitutions  
1152 necessary, it is possible to create a UFiP identifying the Managed Element that is the target of the  
1153 command.

1154 Some SM CLP commands can be invoked against a UFsT. When the target address path terminates in a  
1155 UFsT, the Command Processor interprets the command to be targeted to all Managed Elements of that  
1156 class within that container or uses the class tag as a selector for the action specified by the verb. The  
1157 general syntax of a target address path of this type is as follows:

```
1158 [<address term separator> ] *[ ( "." / ".." / <UFiT> ) <address term separator> ]  
1159 <UFsT>
```

1160 As mentioned previously, the SM CLP supports addressing an instance or instances of an association.  
1161 The general syntax for targeting an association is as follows:

```
1162 [[<address term separator> ] *[ ( "." / ".." / <UFiT> ) <address term separator> ]  
1163 <UFiT>] "="<Association Class>["="<address term separator> ]*( <UFiT> <address  
1164 term separator> ) <UFiT> ]
```

1165 The target address terms leading up to the first occurrence of "=" are used in accordance with the rules  
1166 for generating an instance address path. This instance address path identifies one of the Managed  
1167 Elements referenced by the target association. Following these rules, if no target address terms precede  
1168 the first "=", the CDT will be selected as the referenced instance. The <Association Class>  
1169 identifies the Association Class that will be searched for instances. The second occurrence of "=" and  
1170 following additional target address terms are optional. The second "=" is the ending delimiter of the  
1171 <Association Class>. The target address terms which follow are used to construct an instance  
1172 address path identifying the other Managed Element referenced by the association. If the second set of  
1173 address terms is omitted, the implementation will return all instances of the Association Class that  
1174 reference the instance addressed on the left-hand side.

1175 Using the target notation, the following example target addresses can be constructed:

```
1176 /  
1177 /system1  
1178 \system1  
1179 system1  
1180 /system1/alarm3  
1181 alarm3  
1182 ../rack3  
1183 hw1/../../../rack3  
1184 ..  
1185 .  
1186 /system1=>AssociatedPowerManagementService=>/system1/service24  
1187 ../system1/cpul=>SystemDevice=>/system1
```



1188 `../system1/cpul=>SystemDevice`

### 1189 **5.2.1.3.6 Target Address Evaluation**

1190 CLP commands fall into two categories: commands that accept a command target term and commands  
1191 that do not. Not all commands that accept a command target require one to be included. The rules of  
1192 precedence that govern choosing among an Implicit Command Target, the CDT, and a command target  
1193 term are detailed in 5.1.3.3. This clause describes the rules for evaluating the command target term if it is  
1194 specified.

1195 The command target term can be either a Relative Target Address or an Absolute Target Address. The  
1196 command target term will identify a specific Managed Element, a set of Managed Elements identified by  
1197 their class, a specific association, or the associations of a particular Association Class which reference a  
1198 specific Managed Element.

#### 1199 **5.2.1.3.6.1 Rules for Addressing a Specific Association**

1200 If the command target term includes exactly two occurrences of the association separator, the  
1201 implementation shall evaluate the command target term according to the following rules:

- 1202 • The implementation shall interpret the characters between the two occurrences of the  
1203 association separator as identifying the Association Class.
- 1204 • The implementation shall interpret all characters prior to the first occurrence of the association  
1205 separator as a single token. The implementation shall evaluate the token according to the  
1206 "Rules for Addressing a Target Instance" (5.2.1.3.6.4).
- 1207 • The implementation shall interpret all characters after the second occurrence of the association  
1208 separator as a single token. The implementation shall evaluate the token according to the  
1209 "Rules for Addressing a Target Instance" (5.2.1.3.6.4).

1210 The implementation shall set the Resultant Target for the command to the association instance of the  
1211 type specified by the Association Class such that the association references the Managed Element  
1212 identified by the instance address preceding the first association separator and the association references  
1213 the Managed Element identified by the instance address following the second occurrence of the  
1214 association separator.

#### 1215 **5.2.1.3.6.2 Rules for Addressing Instances of an Association**

1216 If the command target term includes exactly one occurrence of the association separator, the command  
1217 target term is assumed to be targeting all instances of a particular Association Class that reference a  
1218 Managed Element instance, and the implementation shall evaluate the command target term according to  
1219 the following rules:

- 1220 • The implementation shall interpret the characters between the occurrence of the association  
1221 separator and the Command Line term separator as identifying the Association Class.
- 1222 • The implementation shall interpret all characters prior to the first occurrence of the association  
1223 separator as a single instance address. The implementation shall evaluate the instance address  
1224 according to the "Rules for Addressing a Target Instance" (5.2.1.3.6.4).

1225 The implementation shall set the Resultant Target to the set of associations such that for each  
1226 association in the set, the association is of the type specified by the Association Class and references the  
1227 Managed Element identified by the instance address preceding the first occurrence of the association  
1228 separator in the command target term.

#### 1229 **5.2.1.3.6.3 Rules for Addressing a Target Instance/Class**

1230 If the command target term does not include any occurrences of the association separator, the  
1231 implementation shall evaluate the command target term according to the "Rules for Addressing a Target  
1232 Instance" (5.2.1.3.6.4).

## Server Management Command Line Protocol (SM CLP) Specification

### 1233 5.2.1.3.6.4 Rules for Addressing a Target Instance

1234 Implementations shall evaluate instance address terms according to the following rules:

- 1235 • If the instance address term begins with the address term separator, the instance address term  
1236 is considered to be an Absolute Target Address. The implementation shall interpret an Absolute  
1237 Target Address as relative to the Managed Element instance that is the root of the MAP's  
1238 address space.
- 1239 • If the instance address term does not begin with the address term separator, the instance  
1240 address term is considered a Relative Target Address. The implementation shall interpret a  
1241 Relative Target Address as relative to the Current Default Target and shall prepend the instance  
1242 address term with the UFiP of the Current Default Target and an address term separator prior to  
1243 evaluating the instance address term.
- 1244 • Implementations shall evaluate the instance address term from left to right as follows, using the  
1245 address term separator as a token delimiter:
  - 1246 – If the token is a "." (dot), remove the token from the instance address term.
  - 1247 – If the token is a ".." (dot dot), remove the token from the instance address term and  
1248 remove the preceding UFiT, if a preceding UFiT is present.
  - 1249 – If the token is a UFiT, leave it in the instance address term.
  - 1250 – If the token is a UFST, leave it in the instance address term.

1251 After evaluating the instance address term using the preceding rules, the instance address term will be a  
1252 UFiP and is the Resultant Address produced by applying the rules for addressing a target instance. Note  
1253 that after applying these rules, it is possible that the Resultant Address will consist of a single address  
1254 term separator character. Implementations shall interpret this Resultant Address as equivalent to the  
1255 UFiP of the Managed Element that is the root of the address space.

### 1256 5.2.1.3.7 Target Properties

1257 This clause specifies constraints for target properties.

#### 1258 5.2.1.3.7.5 General

1259 Many CLP verbs accept target property terms as input to the command. Target property terms always  
1260 contain a target property name and optionally contain the assignment or equivalence operator followed by  
1261 a property value. Implementations shall interpret terms appearing in the Command Line after the  
1262 command target term as target property terms. Implementations shall interpret target property names in a  
1263 case-insensitive manner.

1264 When the command target term is omitted, implementations shall interpret any non-option name terms as  
1265 target property terms.

1266 When a structured output is specified, the implementation shall return string values for each property  
1267 name and property value such that the implementation will accept the property name and property value  
1268 as input when they are specified according to the rules in "Rules for Specifying Target Property Values"  
1269 (5.2.1.3.7.1). There are three types of target property terms: terms that include the assignment operator,  
1270 terms that include the equivalence operator, and terms that do not include either. Terms that include the  
1271 assignment operator are used to indicate a desired value to assign to a property and are interpreted  
1272 according to "Using the Assignment Operator" (5.2.1.3.7.3). Terms that include an equivalence operator  
1273 are used to indicate a property name and desired value for the property when filtering for an instance with  
1274 that property and are interpreted according to the rules in "Using the Equivalence Operator" (5.2.1.3.7.4).

#### 1275 5.2.1.3.7.1 Rules for Specifying Target Property Values

1276 A CLP implementation will accept target property values as part of a target property term. They can be  
1277 used with some CLP verbs (`create` and `set`) to specify a value to assign to a property or with some

1278 CLP verbs and options (`show` and `display`) to filter results based on a property/value match. When a  
1279 user specifies a target property value on the Command Line, the implementation shall enforce the  
1280 following syntax:

1281 If the property value contains a CLP reserved character, the value is enclosed in quotes. If the  
1282 property value includes a " (double quote) character, the " (double quote) is escaped using the CLP  
1283 escape character.

1284 The specific format of the value for a property is defined in DSP0216. Note that in the case of a property  
1285 that is a Value/ValueMap, the string supplied as a value to the property for assignment could be the string  
1286 representation of the numeric value or the actual value mapped string constant.

### 1287 5.2.1.3.7.2 Rules for Specifying Array Properties

1288 Some properties on Managed Elements are arrays. The CLP provides two methods for dealing with array  
1289 properties. Implementations shall support both methods. The first method allows individual positions  
1290 within an array property to be addressed by index using bracket notation. Bracket notation consists of a  
1291 property name followed by an opening bracket ('['), followed by one or more characters specifying the  
1292 desired index, followed by a closing bracket (']'). Note that no white space occurs anywhere between the  
1293 property name and the closing bracket. When a property target term includes a '[' character, followed by  
1294 a ']' character, the implementation shall interpret the characters that occur between the two brackets as  
1295 specifying the index of the position within the array property that is being addressed. For each array  
1296 property, legal values for the index are defined by the MOF (*4HCommon Information Model (CIM)*  
1297 *Schema, version 2.12*) that defines the class to which the property belongs. The syntax for addressing a  
1298 position within an array property is as follows:

```
1299 <property name>["<index>"]
```

1300 This syntax is supported wherever a property name/value is accepted by the CLP.

1301 The alternate approach is supported only for the assignment of values to an array property. It is  
1302 documented in the following clause. If a client uses array notation with a property that is not an array  
1303 property, the implementation will return an error.

### 1304 5.2.1.3.7.3 Using the Assignment Operator

1305 The assignment operator is used to indicate a desired value to be assigned to a property. The syntax for  
1306 using the assignment operator in a target property term is as follows:

```
1307 <property name>=<property value>
```

1308 When the property name contains the bracket notation defined in "Rules for Specifying Array Properties"  
1309 (5.2.1.3.7.2), the implementation shall assign the property value to the array position identified by the  
1310 index delimited by the brackets.

1311 If the property is multi-valued (an array), multiple array positions can be assigned using a comma-  
1312 delimited list of values. When the target property value of a target property term is a comma-delimited list,  
1313 the implementations shall interpret each comma-delimited token in the target property value as the value  
1314 to be assigned to the corresponding array position of the property. When the comma-delimited token is a  
1315 zero-length string, the implementation shall not assign a value to the corresponding array position. When  
1316 <property name> identifies an array property, and neither of the two methods for managing array  
1317 properties is used, the implementation shall attempt to assign <property value> to the first position in  
1318 the array.

### 1319 5.2.1.3.7.4 Using the Equivalence Operator

1320 The equivalence operator is used to indicate that an implementation filters results for instances that have  
1321 a property with the specified name and value. The syntax for using the equivalence operator is as follows:

```
1322 <property name>===<property value>
```

## Server Management Command Line Protocol (SM CLP) Specification

1323 When the property name contains the bracket notation defined in "Rules for Specifying Array Properties"  
1324 (5.2.1.3.7.2), the implementation shall compare the property value to the value of the array position  
1325 identified by the index delimited by the brackets. When <property name> identifies an array property,  
1326 and the bracket notation defined in "Rules for Specifying Array Properties" is not used, the  
1327 implementation shall compare <property value> to all array positions in the property.

### 1328 5.2.2 Output Data Schema

1329 This clause describes valid data elements and associated values for inclusion in a Command Response.

#### 1330 5.2.2.1 Command Response Organization

1331 In the absence of communication errors or session termination, every CLP command will result in a  
1332 Command Response being returned to the Client. A Command Response consists of Command Status  
1333 data and Command Results data. The Command Response data is ordered as follows:

- 1334 • Command Status data (for example, successful or errors/exceptions)
- 1335 • Command Results data (for example, information generated from/by the command)

1336 See 5.1.10 for full descriptions of the data elements. CLP command options described in Clause 7 control  
1337 the content and format of output data.

#### 1338 5.2.2.2 Common Output Keywords

1339 Common output keywords (Table 2) are those data elements that may appear in any response data.  
1340 Common keywords are used for data organization purposes—identification, grouping, sorting, and so on.

1341 **Table 2 – Common Output Keywords**

Keyword	Definition
association	Indicates that the data is a target association.
endgroup	Indicates the end of a group of output data elements.
group	Indicates the beginning of a new group of output data elements that are to be interpreted as a group.
property, properties	Indicates that the data is a target property name.
target	Indicates that the data is a Managed Element that may contain other Managed Elements.
targets	Indicates that the data lists targets contained in an element.
ufct	Indicates that the data is a User-Friendly class Tag.
ufit	Indicates that the data is a User-Friendly instance Tag.
ufip	Indicates that the data is a User-Friendly instance Path (fully qualified path).
verb, verbs	Indicates that the data is a command verb name.
endoutput	Indicates the end of a "keyword" Command Response.

### 1342 5.2.3 Command Status Data Elements

1343 The following clauses describe the constraints on each of the Command Status data elements.

#### 1344 5.2.3.1 Command Status Keywords

1345 A Command Response includes Command Status data elements.

1346 Table 3 lists the keywords used to identify properties of the Command Status data elements.

1347

**Table 3 – Command Status Keywords**

Keyword	Definition
status	The Status property. This is one of the values defined in Table 4.
status_tag	The status_tag property. This is one of the values defined in Table 4.
error	The Processing Error property detected by the CLP Service. This is one of the integer values in Table 6.
error_tag	The error_tag property. This is the string value in Table 6.

1348 The Command Status indicates the processing disposition of the command entered. When the `status`  
 1349 keyword is returned, implementations shall assign it one of the values listed in Table 4. When the  
 1350 `status_tag` keyword is returned, implementations shall assign it the value in Table 4 that corresponds  
 1351 to the value of the `status` keyword.

1352

**Table 4 – Command Status Values and Tags**

status	status_tag	Description
0	COMMAND COMPLETED	Status = Completed. The command and any associated jobs have completed successfully. The command and any ME jobs completed within command execution. No job remains in-flight and no job ID is active for this command.
1	COMMAND SPAWNED	Status = Spawned. The command returned an interim response to the Client but continues to run as a spawned job. The Job ID of the spawned command may be used to retrieve the Command Status.
2	COMMAND PROCESSING FAILED	Status = COMMAND PROCESSING FAILED. No job was created. No job remains in-flight and no job ID is active for this command.
3	COMMAND EXECUTION FAILED	Status = COMMAND EXECUTION FAILED. The command and any associated jobs ran to completion and failed. The command and any ME jobs completed within command execution.

1353 Table 5 lists the keywords used to identify properties of the Message data element. Each `message_arg`  
 1354 identifies a string value for insertion into the message text. If an implementation supports message  
 1355 argument insertion into message text, the implementation shall identify each insertion location in the  
 1356 message text using the character sequence `{n}`. Implementations shall interpret the value of `n` as  
 1357 identifying the index of the message argument to insert.

1358

**Table 5 – Message Keywords**

Keyword	Definition
message	Message data element—A free-form text explanation of the Command Status or error.
message_id	Message Id data element—A unique text string identifier for the status or error message that can be used by the Client to locate any translations of the message in other languages.
message_arg	Message Argument data element—Substitution value for insertion into a message.
owningentity	Owning Entity data element—A unique string identifier for the owner of the message identifier. The owning entity and message id combine to form a unique key for looking up message text translations.

## Server Management Command Line Protocol (SM CLP) Specification

1359 Table 6 lists the valid values for the `error` and `error_tag` keywords. When an implementation includes  
 1360 the `error` and `error_tag` keywords in a Command Response, the implementation shall assign them  
 1361 values from Table 6.

1362 **Table 6 – Processing Error Values and Tags**

error	error_tag	Description
255	COMMAND ERROR – UNSPECIFIED	Unspecified command error; used only when other command errors are not applicable.
254	COMMAND NOT SUPPORTED	The command is recognized as a CLP command verb but is not supported by this implementation.
253	COMMAND NOT RECOGNIZED	The command is syntactically correct, but the implementation does not recognize the first term in the command as a verb (that is, cannot report "not supported" because the verb is unknown to the implementation).
252	COMMAND SYNTAX ERROR	The command is recognized as a CLP command verb, but the syntax has not been correctly followed.
251	INVALID OPTION	The command is recognized as a CLP command verb, the syntax is correct, but an option is not valid.
250	INVALID ARGUMENT	The command is recognized as a CLP command verb, the syntax is correct, but an argument value for an option is not valid.
249	OUTPUT FORMAT NOT SUPPORTED	The user selected an output format that is not supported by this implementation.
248	MISSING ARGUMENT	The command is recognized as a CLP command verb, the syntax is correct, but an argument value for an option is missing.
247	OPTION NOT SUPPORTED	The command is recognized as a CLP command verb, the syntax is correct, but an option is not supported.
246	INVALID TARGET	The first non-option or option argument term after the verb contained a CLP addressing character but did not adhere to the CLP command target term syntax.
245	REQUIRED OPTION MISSING	The specified command requires an option that was not supplied.
244	QUEUE FULL	A job cannot be started to execute the command.
243	UNRECOGNIZED OEM EXTENSION	The Command Line includes an OEM Extension Name String that is unrecognized by the implementation.
242	MISSING REQUIRED TARGET	The command verb requires that a command target term be specified to identify a specific target for the command, and a command target term was not included in the Command Line.
241	FUNCTION NOT SUPPORTED	The command syntax is valid but included a request for optional behavior that is not supported by this implementation.

1363 When an error occurs processing a command prior to creating a job to execute the command and this  
 1364 specification does not identify a specific Processing Error to use to indicate the error condition, the  
 1365 implementation shall return a Command Status of `COMMAND PROCESSING FAILED` and a Processing  
 1366 Error of `COMMAND ERROR – UNSPECIFIED`.

### 1367 5.2.3.2 Job Error Keywords

1368 When the Command Status is `COMMAND EXECUTION FAILED`, a Job Error will follow the Command  
 1369 Status to describe the details of the failure. Table 7 defines the valid keywords and values for the Job  
 1370 data element. The accepted values and corresponding descriptions for each value are provided in the  
 1371 tables that follow. For each keyword, implementations shall return data that conforms to the restrictions  
 1372 specified for the keyword in Table 7.

1373

Table 7 – Job Error Keywords

Keyword	Definition
job_id	Job Identifier—An integer value in the range [1, 65535] inclusive.
errtype	Execution Error—Provides the primary classification of the error.
errtype_desc	Execution Error Tag—The character string tag corresponding to the Execution Error.
cimstat	CIM Status—A value that describes the error as it relates to the CIM Server.
cimstat_desc	An enumerated string, corresponding to the value of cimstat.
severity	A value that describes the severity of the error from the notifier's point of view.
severity_desc	An enumerated string, corresponding to the value of severity.
probcause	A value that describes the probable cause of the error.
probcause_desc	An enumerated string, corresponding to the value of probcause.
recmdaction	A free-form string that describes a recommended action. Zero or more recommended actions appear per Job Error occurrence.
errsource	A string that identifies the Managed Element generating this Job Error instance.
errsourceform	A value that identifies the format of the error source string identifier.
errsourceform_desc	A free-form string describing and corresponding to the error source format.

1374 The Execution Error property communicates the primary category of the Job Error. If an implementation  
 1375 includes the `errtype` keyword in a Command Response, the implementation shall assign the keyword  
 1376 one of the values from Table 8. If an implementation includes the `errtype_desc` keyword in a  
 1377 Command Response, the implementation shall assign the keyword the value from Table 8 that  
 1378 corresponds to the value assigned to the `errtype` keyword. The values for `errtype` and  
 1379 `errtype_desc` correspond to the ValueMap and Values for the ErrorType property of CIM\_Error.

1380

Table 8 – Error Type Values and Descriptions

errtype	errtype_desc	Description
0	Unknown	None
1	Other	None
2	Communications Error	The command or operation cannot be initiated because the ME is not responding. or The job is terminated because the target ME is not responsive and the MAP cannot determine the progress of the operation. Note that the state change activity may still be in-progress at the ME but the implementation cannot communicate with the ME to determine the status.
3	Quality of Service Error	None
4	Software Error	None
5	Hardware Error	None
6	Environmental Error	None
7	Security Error	None
8	Oversubscription Error	None
9	Unavailable Resource Error	CLP Service cannot acquire needed internal resources to process the command.
10	Unsupported Operation Error	None

## Server Management Command Line Protocol (SM CLP) Specification

1381 The CIM Status property communicates any management layer or instrumentation layer errors  
 1382 encountered by the CLP Service in its attempt to initiate the requested operations for the specified  
 1383 targets. Errors that occur when attempting to set the value for a property result in one of the errors listed  
 1384 in Table 9 being returned. If an implementation includes the `cimstat` keyword in a Command Response,  
 1385 the implementation shall assign the keyword one of the values from Table 9. If an implementation  
 1386 includes the `cimstat_desc` keyword in a Command Response, the implementation shall assign the  
 1387 `cimstat_desc` keyword the value from Table 9 that corresponds to the value assigned to the `cimstat`  
 1388 keyword. The values for `cimstat` and `cimstat_desc` correspond to the ValueMap and Values for the  
 1389 CIMStatusCode property of CIM\_Error.

1390

**Table 9 – CIM Status Code Values and Descriptions**

<code>cimstat</code>	<code>cimstat_desc</code>	Description
1	CIM_ERR_FAILED	A general, unspecified error occurred.
2	CIM_ERR_ACCESS_DENIED	The user does not have proper authorization to use the command. or The command was authorized for the user, but the user is not authorized to perform a resulting operation on this or a dependent target ME. or The user does not have access to a CIM resource.
3	CIM_ERR_INVALID_NAMESPACE	The target namespace does not exist.
4	CIM_ERR_INVALID_PARAMETER	The verb is recognized, the command syntax is correct, option names are correct, but an option argument value is not valid. One or more target property values or option argument values that specify target properties are invalid.
5	CIM_ERR_INVALID_CLASS	The class indicated by the UFcT or the Association Class does not exist in the scope of the command.
6	CIM_ERR_NOT_FOUND	The command target is not found. The requested UFiT could not be found or was unresponsive.
7	CIM_ERR_NOT_SUPPORTED	The command is valid but the target specified does not support the necessary operation or operations needed to carry out the command. OR The user has requested an operation that is not supported by this target ME.
8	CIM_ERR_CLASS_HAS_CHILDREN	The operation cannot be carried out on this class because it has subclasses with instances.
9	CIM_ERR_CLASS_HAS_INSTANCES	The operation cannot be carried out on this class because it has instances.
10	CIM_ERR_INVALID_SUPERCLASS	The operation cannot be carried out because the superclass does not exist.
11	CIM_ERR_ALREADY_EXISTS	The operation cannot be carried out because the specified UFiT already exists.



## Server Management Command Line Protocol (SM CLP) Specification

cimstat	cimstat_desc	Description
12	CIM_ERR_NO_SUCH_PROPERTY	The specified Property does not exist for the command target.
13	CIM_ERR_TYPE_MISMATCH	The value supplied for a property is incompatible with the property's data type.
14	CIM_ERR_QUERY_LANGUAGE_NOT_SUPPORTED	(RESERVED FOR FUTURE USE)
15	CIM_ERR_INVALID_QUERY	(RESERVED FOR FUTURE USE)
16	CIM_ERR_METHOD_NOT_AVAILABLE	The extrinsic Method could not be executed for the command target. or An operation currently executing on the Target is performing an internal function such that the requested operation cannot be concurrently executed. or The target ME is in use by another session.
17	CIM_ERR_METHOD_NOT_FOUND	The user has requested an operation that is not recognized by the target ME. The specified extrinsic method could not be found for the command target.
18	CIM_ERR_UNEXPECTED_RESPONSE	The returned response from the target was unexpected. or The job spawned by the previous command has ended prematurely and failed. For example, a firmware update may abort if retrieval of an image from a URI times out.
19	CIM_ERR_INVALID_RESPONSE_DESTINATION	(RESERVED FOR FUTURE USE)
20	CIM_ERR_NAMESPACE_NOT_EMPTY	(RESERVED FOR FUTURE USE)

1391 The Severity property communicates the urgency of the error, from the MAP's perspective. If an  
 1392 implementation includes the `severity` keyword in a Command Response, the implementation shall  
 1393 assign the keyword one of the values from Table 10. If an implementation includes the `severity_desc`  
 1394 keyword in a Command Response, the implementation shall assign the `severity_desc` keyword the  
 1395 value from Table 10 that corresponds to the value assigned to the `severity` keyword. The values for  
 1396 `severity` and `severity_desc` correspond to the ValueMap and Values for the PerceivedSeverity  
 1397 property of CIM\_Error. "1" is not specified in the ValueMap and therefore is reserved by the CLP.

1398

**Table 10 – Severity Values and Descriptions**

severity	severity_desc	Description
0	Unknown	The severity is unknown or unassigned by the implementation.
1	Reserved	This value is reserved and should not be used.
2	Low	Used for non-critical issues such as invalid parameters, incorrect usage, and unsupported functionality.
3	Medium	Used to indicate action is needed, but the situation is not serious at this time.
4	High	Used to indicate action is needed immediately.
5	Fatal	Used to indicate a loss of data or unrecoverable system or service failure.

## Server Management Command Line Protocol (SM CLP) Specification

1399 The Probable Cause and Probable Cause Description properties identify the probable cause of an  
 1400 execution error. Any errors generated by the Managed Element itself are characterized in the Probable  
 1401 Cause property, described in Table 11. If an implementation includes the `probcause` keyword in a  
 1402 Command Response, the implementation shall assign the keyword one of the values from Table 11. If an  
 1403 implementation includes the `probcause_desc` keyword in a Command Response, the implementation  
 1404 shall assign the `probcause_desc` keyword the value from Table 11 that corresponds to the value  
 1405 assigned to the `probcause` keyword. The values for `probcause` and `probcause_desc` correspond to  
 1406 the ValueMap and Values for the ProbableCause property of `CIM_Error`.

1407 **Table 11 – Probable Cause Values and Descriptions**

probcause	probcause_desc
0	Unknown
1	Other
2	Adapter/Card Error
3	Application Subsystem Failure
4	Bandwidth Reduced
5	Connection Establishment Error
6	Communications Protocol Error
7	Communications Subsystem Failure
8	Configuration/Customization Error
9	Congestion
10	Corrupt Data
11	CPU Cycles Limit Exceeded
12	Dataset/Modem Error
13	Degraded Signal
14	DTE-DCE Interface Error
15	Enclosure Door Open
16	Equipment Malfunction
17	Excessive Vibration
18	File Format Error
19	Fire Detected
20	Flood Detected
21	Framing Error
22	HVAC Problem
23	Humidity Unacceptable
24	I/O Device Error
25	Input Device Error
26	LAN Error
27	Non-Toxic Leak Detected
28	Local Node Transmission Error
29	Loss of Frame
30	Loss of Signal
31	Material Supply Exhausted
32	Multiplexer Problem
33	Out of Memory

## Server Management Command Line Protocol (SM CLP) Specification

probcause	probcause_desc
34	Output Device Error
35	Performance Degraded
36	Power Problem
37	Pressure Unacceptable
38	Processor Problem (Internal Machine Error)
39	Pump Failure
40	Queue Size Exceeded
41	Receive Failure
42	Receiver Failure
43	Remote Node Transmission Error
44	Resource at or Nearing Capacity
45	Response Time Excessive
46	Retransmission Rate Excessive
47	Software Error
48	Software Program Abnormally Terminated
49	Software Program Error (Incorrect Results)
50	Storage Capacity Problem
51	Temperature Unacceptable
52	Threshold Crossed
53	Timing Problem
54	Toxic Leak Detected
55	Transmit Failure
56	Transmitter Failure
57	Underlying Resource Unavailable
58	Version Mismatch
59	Previous Alert Cleared
60	Login Attempts Failed
61	Software Virus Detected
62	Hardware Security Breached
63	Denial of Service Detected
64	Security Credential Mismatch
65	Unauthorized Access
66	Alarm Received
67	Loss of Pointer
68	Payload Mismatch
69	Transmission Error
70	Excessive Error Rate
71	Trace Problem
72	Element Unavailable
73	Element Missing
74	Loss of Multi Frame
75	Broadcast Channel Failure
76	Invalid Message Received

## Server Management Command Line Protocol (SM CLP) Specification

probcause	probcause_desc
77	Routing Failure
78	Backplane Failure
79	Identifier Duplication
80	Protection Path Failure
81	Sync Loss or Mismatch
82	Terminal Problem
83	Real Time Clock Failure
84	Antenna Failure
85	Battery Charging Failure
86	Disk Failure
87	Frequency Hopping Failure
88	Loss of Redundancy
89	Power Supply Failure
90	Signal Quality Problem
91	Battery Discharging
92	Battery Failure
93	Commercial Power Problem
94	Fan Failure
95	Engine Failure
96	Sensor Failure
97	Fuse Failure
98	Generator Failure
99	Low Battery
100	Low Fuel
101	Low Water
102	Explosive Gas
103	High Winds
104	Ice Buildup
105	Smoke
106	Memory Mismatch
107	Out of CPU Cycles
108	Software Environment Problem
109	Software Download Failure
110	Element Reinitialized
111	Timeout
112	Logging Problems
113	Leak Detected
114	Protection Mechanism Failure
115	Protecting Resource Failure
116	Database Inconsistency
117	Authentication Failure
118	Breach of Confidentiality
119	Cable Tamper

probcause	probcause_desc
120	Delayed Information
121	Duplicate Information
122	Information Missing
123	Information Modification
124	Information Out of Sequence
125	Key Expired
126	Non-Repudiation Failure
127	Out of Hours Activity
128	Out of Service
129	Procedural Error
130	Unexpected Information

1408 **5.2.4 Output Data Formats**

1409 The CLP specifies the following named, selectable formats for output data: "text", "keyword", and  
1410 "clpxml". These data formats are defined in the following clauses.

1411 **5.2.4.1 General**

1412 Implementations shall support "text" format and shall provide "text" format output as the default output  
1413 setting. When other output data formats are supported, implementations shall allow the user to override  
1414 the format on a per-command basis using the command option `output`. Implementations shall also  
1415 provide an environment setting to control output format for all commands, unless overridden by the user.  
1416 If an implementation supports at least one output format other than "text", the implementation shall  
1417 support the "clpxml" output format.

1418 **5.2.4.2 Text Format**

1419 The output format "text" is the default format for output. Output in "text" format is not recommended to be  
1420 parsed by an automated agent. This format is suitable only to be read by a person. Text output format will  
1421 vary from implementation to implementation.

1422 When "text" output format is selected, implementations may use any output text wording that is deemed  
1423 appropriate to convey the Command Response data elements to the user. When the Command  
1424 Response is presented in "text" format, the implementation may provide execution status data as part of  
1425 the text description of the Command Results or, if the command is successful, the implementation may  
1426 not include an explicit statement of execution status in the Command Response.

1427 **5.2.4.3 Structured Outputs**

1428 This clause details requirements related to structured output.

1429 **5.2.4.3.1 General**

1430 The CLP specification defines two structured output formats: "keyword" and "clpxml". When returning a  
1431 Command Response formatted according to a structured output, the implementation shall use the specific  
1432 keywords and values identified in 5.2.3.1 and 5.2.3.2 for each data element included in the Command  
1433 Status data element.

1434 For information about the rules governing the use of the `output` option to select an output format, see  
1435 **Error! Reference source not found..**

## Server Management Command Line Protocol (SM CLP) Specification

### 1436 5.2.4.3.2 Keyword=Value Format

1437 The "keyword" output format requests the command to format the output in a "keyword=value" format. To  
1438 select "keyword" format explicitly, the implementations shall accept "keyword" as the argument value for  
1439 the format argument to the output option.

1440 In "keyword" output format, output data element items appear in sequence as "<keyword>=<value>"  
1441 items separated by the end-of-line character sequence. Implementations shall use double quotes around  
1442 any value that contains the end-of-line sequence.

1443 Implementations shall specify a group of "keyword" items that are to be interpreted as a single item or  
1444 collection by using the "begingroup" keyword and a value identifying the type of data. Implementations  
1445 shall terminate a group by using the "endgroup" keyword. When a "begingroup" keyword appears in the  
1446 output, all keywords that follow are interpreted as part of a group until the next "endgroup" keyword.  
1447 Implementations shall indicate the end of the Command Response by using the "endoutput" keyword.  
1448 Implementations may include blank lines or lines that have an # character (octothorp) in the first character  
1449 position in the output. If an implementation includes blank lines or lines that have an # character  
1450 (octothorp) in the first character position in the output, the implementation shall not impart a meaning to  
1451 the blank lines.

1452 The general form of the "keyword" output format is as follows:

```
1453 commandline=the commandline that was processed
1454 status=Integer job status code
1455 status_tag=String job status
1456 error=integer processing error code, if there is one
1457 error_tag=string description of processing error
1458 begingroup=message
1459 owningentity=organization owning a message
1460 message_id=string identifier for the message, unique with the value of owningentity
1461 message=message text
1462 message_arg=Insertion value 1 for the message
1463 .
1464 .
1465 .
1466 message_arg=Insertion value n for the message
1467 endgroup
1468 job_id=Identifier for the job created to execute the command
1469 errtype=Integer code for the high level category of execution error
1470 errtype_desc=string description of the execution error
1471 cimstat=integer code for the CIM error
1472 cimstat_desc=string description of the CIM error
1473 severity=integer code indicating the severity of the error
1474 severity_desc=description corresponding to the severity code
1475 probcause=integer code indicating the probable cause of the execution error
1476 probcause_desc=description corresponding to the probable cause code
1477 errsource=Target Address of error source
1478 errsourceform=SMA Target Address
1479 errsourceform_desc=SM Target Address
1480 recmdaction=Free-form string describing action to take to resolve the error
1481 begingroup=message
1482 owningentity=organization owning a message
1483 message_id=string identifier for the message, unique with the value of owningentity
1484 message=message text
1485 message_arg=Insertion value 1 for the message
1486 .
1487 .
1488 .
1489 message_arg=Insertion value n for the message
1490 endgroup
```

```

1491 .
1492 .
1493 .
1494 command=<verbname>
1495 .
1496 .
1497 .
1498 Verb and target-specific keywords
1499 .
1500 .
1501 .
1502 endoutput

```

1503 If an implementation includes OEM output for a command issued with a CLP verb, the implementation  
1504 shall return the OEM output after the standard output for the command and before the final "endoutput"  
1505 keyword. If an implementation includes OEM defined keywords for inclusion in the output, the  
1506 implementation shall define each keyword using the convention for OEM name extensions defined in  
1507 5.2.6.2.

1508 If an implementation is returning "keyword" output for an OEM Command Form command, the  
1509 implementation shall return the Command Status using the standard keyword structure and shall  
1510 substitute the "command" keyword and subsequent output with the keyword "oemcommand" followed by  
1511 the vendor-defined output.

### 1512 5.2.4.3.3 XML Format

1513 The output format "clpxml" requests the command to format the output in an XML document format. To  
1514 select "clpxml" format explicitly, implementations shall accept "clpxml" as the value for the `format`  
1515 argument to the `option` option.

1516 In "clpxml" output format, the output data is a well-formed XML document. The XML document schema  
1517 (tags and so on) is defined per command. An outline of the XML document specific to each CLP verb is  
1518 located in the "XML Output" subclause for that verb in Clause 6.

1519 The XML schema defining the Command Response data element is defined using XSD in *Server*  
1520 *Management Command Line Protocol (SM CLP) Command Response XML Schema v1.0* (DSP0224).  
1521 The XML schema is intended to address the requirements of users of the CLP for a simple, parsable  
1522 schema to represent CLP output. It is not intended as a data exchange format to fully represent a CIM  
1523 instance or class. If an implementation returns Command Response data as an XML document, the  
1524 implementation shall ensure that the document default namespace is:

1525 `"http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"`

1526 OEM vendors may extend the Command Response schema. If OEM vendors extend the Command  
1527 Response schema, the implementation shall place the OEM extensions into a distinct namespace and  
1528 shall define a namespace prefix that follows the convention for OEM name extensions defined in 5.2.6.2.

## 1529 5.2.5 Internationalization/Localization

1530 This clause outlines the support for internationalization and localization provided for by the CLP  
1531 specification. For the purposes of the CLP specification, internationalization is interpreted to mean the  
1532 substitution of strings in one language for strings having equivalent meaning in another language.  
1533 Localization refers to the formatting of information for conformance with the norms of a particular locale.

### 1534 5.2.5.1 Command Input

1535 CLP implementations shall not provide support for internationalization of Command Line terms. CLP  
1536 implementations may provide support for localization of input data. Furthermore, a CLP implementation  
1537 shall not support alternative strings for CLP command verbs, option names, and target property names  
1538 except as OEM extensions (see 5.2.6). Commands written using alternative strings for these Command  
1539 Line terms will not be portable from implementation to implementation.

## Server Management Command Line Protocol (SM CLP) Specification

### 1540 5.2.5.2 Command Output

1541 This clause details requirements related to localized command output.

#### 1542 5.2.5.2.1 CLP Service-Side Localization

1543 CLP implementations may support localized CLP command output. If the implementation supports  
1544 localized output, the implementation shall support the `language` session setting and follow the described  
1545 use of the setting as given in **Error! Reference source not found..**

#### 1546 5.2.5.2.2 Client-Side Localization

1547 It is possible that a CLP implementation will support localization of CLP command output by the Client. To  
1548 support localization of output data at the Client, a CLP implementation could support the following  
1549 capabilities:

- 1550 • at least one of the structured output modes (see 5.2.4.3)
- 1551 • capability to report a Message Owner and Message Identifier for each translatable message  
1552 when a structured output mode is selected (see 5.2.2)

### 1553 5.2.5.3 Locale

1554 The CLP does not specify a mechanism for setting a locale in the environment in order to perform  
1555 translations of units of data. Implementations are expected to manage establishment of data units  
1556 (measures, date and time, and so on) through Managed Element settings.

1557 Implementations shall include the appropriate units designation in "text" and structured output formats for  
1558 each Managed Element property returned. This provides the Client the information needed to perform any  
1559 translation of units locally.

1560 OEMs may provide extensions to the standard unit designations per the OEM extensions described in the  
1561 next clause.

## 1562 5.2.6 OEM Extensions

1563 The CLP allows an OEM to add support for vendor-unique commands and output data. This clause  
1564 details requirements related to OEM extensions to the CLP.

### 1565 5.2.6.1 General

1566 A vendor may extend the CLP in the following ways:

- 1567 • by providing OEM commands that conform to one of the specified CLP Extended Forms
- 1568 • by providing OEM output keywords

### 1569 5.2.6.2 OEM Extension Name Strings

1570 Implementations shall identify any OEM Extension Name Strings used for CLP Command Line terms with  
1571 the CLP standard prefix "OEM" followed by a vendor-unique identification string so that they will exist in a  
1572 namespace separate from those that are specified by this document. Conversely, implementations shall  
1573 not support any other command verbs other than those specified in this specification or those identified as  
1574 vendor-specific using an OEM Extension Name String as documented here.

1575 Implementations shall use a value for the OEM string portion of the prefix that uniquely identifies the  
1576 entity that owns and defines the command. The string shall include a copyrighted, trademarked, or  
1577 otherwise unique name that is owned by the business entity or standards body defining the command.

1578 Implementations shall interpret and recognize the standard portion of the string, "OEM", and the vendor  
1579 identifier string as case insensitive.



## Server Management Command Line Protocol (SM CLP) Specification

1580 EXAMPLE ("OEM"=="oem"=="Oem" and "VENDOR"=="vendor"=="Vendor"), where the term "vendor" is not  
1581 taken to be a literal and instead is a value such as "Acme".

### 1582 5.2.6.3 Command Extension Forms

1583 The CLP recognizes two forms of command extension:

- 1584 • CLP Verb Extended Form
- 1585 • OEM Command Line Extended Form

#### 1586 5.2.6.3.1 General

1587 If a Command Line contains an OEM Extension Name String that is not supported by the implementation,  
1588 the implementation shall return a Command Status of COMMAND PROCESSING FAILED and a  
1589 Processing Error of UNRECOGNIZED OEM EXTENSION.

#### 1590 5.2.6.3.2 CLP Verb Extended Form

1591 The CLP Verb Extended Form requires that a standard CLP command verb appear as the first term on  
1592 the Command Line.

##### 1593 5.2.6.3.2.1 General

1594 A vendor may define vendor-specific Command Line terms for options, option arguments, option  
1595 argument values, target addresses, and target properties, as long as those terms follow the semantics  
1596 defined in the CLP specification.

##### 1597 5.2.6.3.2.2 Terms

- 1598 • CLP Verb/Options and/or Option Arguments
- 1599 • OEM Options and/or Option Argument(s)
- 1600 • OEM Target Addresses and/or Property Names

##### 1601 5.2.6.3.2.3 Syntax

```
1602 <CLP verb> *["-"<CLP option> [OEM<vendor><arg name string>]] `
1603 *[-OEM<vendor><optionname> [OEM<vendor><arg name string>]] `
1604 OEM<vendor><target address string>
1605 `
1606 *(OEM<vendor><property name string>)
```

##### 1607 5.2.6.3.2.4 Rules

1608 The CLP specification defines the behavior of the verb and associated CLP options when the option is  
1609 specified with a CLP-defined argument.

- 1610 • OEM Arguments to SM CLP options and property names and values shall observe CLP syntax  
1611 and delimiter rules.
- 1612 • The implementation shall not support OEM option arguments or option argument values that are  
1613 inconsistent with the behavior of the CLP option. The behavior of the OEM-defined argument is  
1614 vendor specific. For example, an OEM argument to the `display` option cannot be used to  
1615 modify the targets of a command.
- 1616 • Implementations shall not accept a short form for an OEM Option.
- 1617 • Implementations of OEM targets/properties shall observe/adhere to the specified CLP  
1618 verb/option behaviors.

## Server Management Command Line Protocol (SM CLP) Specification

- 1619           • When defining OEM target name addresses, implementations shall observe the CLP command  
1620           delimiter characters and may follow CLP target naming addressing syntax or semantics.

1621 The behavior of OEM-defined options is outside the scope of this specification. Vendors are free to define  
1622 the format of arguments to OEM options as their needs dictate. This specification places no restrictions  
1623 on whether options are separated from their arguments by a delimiter, whether options conditionally  
1624 accept arguments, and so on. Therefore, when an OEM-defined option is included in a command, it is  
1625 likely to be necessary to have a priori knowledge of the option in order to deterministically parse the  
1626 Command Line.

### 1627 5.2.6.3.3 OEM Command Line Extended Form

1628 OEM Command Line Extended Form, or OEM Command Form, allows a vendor to provide access to  
1629 vendor-specific commands and command formats. OEM Command Form is indicated by an OEM  
1630 Extension Name String as the first term on the Command Line. This term signals a fully OEM-defined  
1631 command format. Other than this requirement, the commands specified in OEM commands space,  
1632 arguments, and so on are suggested to remain in line with those presented in the CLP specification, but  
1633 are not controlled or defined in any way by this document.

#### 1634 5.2.6.3.3.1 Terms

- 1635           • Full OEM-specified command format  
1636           • Includes form where an OEM extension appears in every CLP Command Line term position

#### 1637 5.2.6.3.3.2 Syntax

```
1638 OEM<vendor> <vendor-specified command line syntax>  
1639 OEM<vendor><verb> <vendor-specified command line syntax>
```

1640 Note the CLP term separator after the first term.

#### 1641 5.2.6.3.3.3 Rules

1642 The vendor completely defines the command syntax, behavior, target addressing, and so on that appear  
1643 after the first term, where the first term is prefixed by "OEM".

### 1644 5.2.6.4 Output Extensions

1645 This clause details requirements related to vendor extensions to the CLP output.

#### 1646 5.2.6.4.1 Vendor-Specific Keywords

1647 A vendor may supply additional output data elements in the response to any CLP command. An  
1648 implementation may support vendor-supplied keyword names. The implementation shall define any  
1649 vendor-supplied keyword names such that they comply with the rules for defining OEM Extension Name  
1650 Strings defined in 5.2.6.2.

1651           EXAMPLE    If vendor "ZYX" introduced an output data element keyword "foobar", the resulting keyword would  
1652           be "OEMZYXfoobar".

#### 1653 5.2.6.4.2 Vendor-Specific Messages and Message Files

1654 Vendors may define and identify vendor-specific messages using the standard SM CLP message  
1655 keywords `message_id`, `message_arg`, and `owningentity` as defined in 5.2.2.

1656 While the keywords and schema for command output are defined by SM CLP, the format of any message  
1657 files local to the Client is outside the scope of this specification.

1658 **6 SM CLP Verbs**

1659 This clause gives a listing of all the verbs supported by the CLP, requirements for support by  
 1660 implementations, and a short description of their basic functionality. The subclauses further define the  
 1661 behavior of each verb.

1662 The documentation for each verb includes a statement of the command line syntax, a description of  
 1663 applicable targets of the verb, and a definition of the command output content, including output keywords  
 1664 to be included in structured forms of output. Where the effect of a supported option is unique to the verb,  
 1665 it will also be explicitly described. For a complete list of options, including which verbs they are supported  
 1666 with, see Clause 7.

1667 Examples are for information only. If an example contradicts specification text elsewhere in the document,  
 1668 the specification text is the authority. General rules and requirements for the Command Response are  
 1669 specified in 5.1.10. See Annex E for the conventions used in the examples.

1670 **6.1 Verb Support Requirements**

1671 The requirements in Table 12 are interpreted as follows:

1672 **shall** If the "Requirement" column in Table 12 contains "shall", implementations shall support  
 1673 the verb specified in the "Command" column of that row.

1674 **PROFILE** If the "Requirement" column in Table 12 contains "PROFILE", implementations will  
 1675 support the verb specified in the "Command" column of that row when the  
 1676 implementations support a profile that defines a mapping for the verb. Specific  
 1677 by-target requirements are found in 32HDSP0216.

1678 The SM CLP syntax and semantics can be comprehended by a human user without intimate knowledge  
 1679 of the CIM Schema. The command verbs, options, targets, and properties are described in a traditional  
 1680 "man page" format, complete with command execution status and output data element descriptions.  
 1681 However, in order to implement a CLP Service, a developer needs to know the mapping of CLP verb  
 1682 option/target/property combinations to the CIM Schema. This mapping information is not included in this  
 1683 specification but is collected in a separate specification as noted above.

1684 If a CLP verb is specified as the first term of a Command Line and the implementation does not support  
 1685 any profiles which require that the verb be supported, the implementation shall not execute the command  
 1686 and the implementation shall return a Command Status of COMMAND PROCESSING FAILED and a  
 1687 Processing Error of COMMAND NOT SUPPORTED.

1688 **Table 12 – Verb Support Requirements**

Command	Requirement	Definition and Usage
cd	shall	Used to set the Current Default Target (navigate the target address space of the MAP).
create	PROFILE	Used to create new instances and associations in the address space of the MAP. This command is allowed only for specific target object types as defined by the profiles and specific MAP implementation.
delete	PROFILE	Used to destroy instances in the address space of the MAP. This command is allowed only for specific target object types as defined by the profiles and specific MAP implementation.
dump	PROFILE	Used to move a binary image from the MAP to a URI.
exit	shall	Used to terminate a CLP session.
help	shall	Used to get context-sensitive help.
load	PROFILE	Used to move a binary image to the MAP from a URI.

## Server Management Command Line Protocol (SM CLP) Specification

Command	Requirement	Definition and Usage
reset	PROFILE	Used to cause a target with power/process control to cycle states from enabled to disabled and back to enabled.
set	shall	Used to set a property or set of properties to a specific value.
show	shall	Used to show values of a property or contents of a collection/target.
start	PROFILE	Used to cause a target with power/process control to change states to a higher run level.
stop	PROFILE	Used to cause a target with power/process control to change states to a lower run level.
version	shall	Used to query the version of the CLP implementation (by default) and other CLP elements (when specified).

### 1689 6.2 cd

1690 The general form of the `cd` command is:

1691 `cd [<options>] [<target>]`

#### 1692 6.2.1 General

1693 For the `cd` command, implementations shall support the syntax defined for the `cd-cmd` term in the CLP  
1694 grammar defined in Annex A.

1695 The `cd` (change default target) command is used to navigate the target address space of the  
1696 implementation. The command changes the Current Default Target for the session. The new target  
1697 address path is specified on the Command Line using the standard CLP target syntax and evaluated  
1698 using the target address evaluation rules. An implementation shall accept a command target term that is  
1699 either an Absolute Target Address or a Relative Target Address. As a result, the command supports both  
1700 relative and absolute path changes. If a command target term is specified, implementations of the `cd`  
1701 command shall evaluate the command target term to a UFiP, validate that the UFiP references a  
1702 Managed Element in the address space of the MAP, and assign the CDT property of the Managed  
1703 Element referenced by SESSION to the UFiP. If the command target term does not evaluate to a UFiP  
1704 according to the rules defined in 5.1.3.5, implementations shall not change the Current Default Target and  
1705 shall return a Command Status of COMMAND PROCESSING ERROR and a Processing Error of  
1706 INVALID TARGET.

1707 Implementations of the `cd` command shall support usage without an argument. If no arguments appear  
1708 on the Command Line, the command shall not change the Current Default Target, the implementation  
1709 shall not attempt to validate that the CDT addresses a valid Managed Element, and the implementation  
1710 shall return the Current Default Target path as output.

1711 Because the CLP supports relative addressing using the reserved character sequence `..`, it is possible  
1712 to construct a target address that nominally references a point beyond the beginning of the session's root  
1713 administration domain. Applying the command target term evaluation rules defined in 5.2.1.3.6 will result  
1714 in the command target term being resolved to the root of the session's address space. Thus an attempt to  
1715 use the `cd` command to change the Current Default Target to a point beyond the beginning of address  
1716 space will result in the CDT being assigned to the session's root administration domain.

#### 1717 6.2.2 Valid Targets

1718 Implementations of the `cd` command will accept an Absolute or a Relative Target Address for the  
1719 command target term. The `cd` command has an Implicit Command Target, which is the session to which  
1720 the SESSION Reserved Target will resolve.

1721 **6.2.3 Options**

1722 Implementations of the `cd` command support the options specified in **Error! Reference source not**  
 1723 **found..**

1724 **6.2.4 Output**

1725 This clause details the requirements for output of the `cd` verb.

1726 **6.2.4.1 Text Format**

1727 The Command Results data shall include the Current Default Target that is in effect when the command  
 1728 completes.

1729 If the implementation cannot determine the current target address (due to error), the implementation shall  
 1730 return text indicating that the Current Default Target is invalid.

1731 **6.2.4.2 Structured Format**

1732 This clause details requirements for structured output formats for the `cd` verb.

1733 **6.2.4.2.1 General**

1734 The returned data shall include any status data in the standard format at the top of the response.

1735 **6.2.4.2.2 XML Output**

1736 The implementation shall return the `cd` element in the `response` element as defined in the Command  
 1737 Response schema in DSP0224.

```
1738 <cd>
1739   <ufip> User Friendly instance Path of the CDT </ufip>
1740 </cd>
```

1741 **6.2.4.2.3 Keyword**

1742 Implementations shall use the following form when returning Command Results for the `cd` command in  
 1743 "keyword" format.

```
1744 command=cd
1745   ufip=User Friendly instance Path of the CDT
1746 endoutput
```

1747 **6.2.5 Examples**

1748 The following examples assume that the user is on a four-CPU system and each command starts with the  
 1749 Current Default Target set to `/system1/cpu2`.

1750 EXAMPLE 1: Returns the Current Default Target and does not change it. No validation of the CDT is performed  
 1751 by the implementation.

```
1752 -> cd
1753 /system1/cpu2
```

1754 EXAMPLE 2: Returns the Current Default Target and does not change it. The CDT is validated and appropriate  
 1755 Command Response data is returned if it is no longer valid.

```
1756 -> cd .
1757 /system1/cpu2
```

## Server Management Command Line Protocol (SM CLP) Specification

1758	EXAMPLE 3:	Moves to the parent (container) of the Current Default Target (up the tree) one level.
1759		<pre>-&gt; cd ..</pre>
1760		<pre>/system1</pre>
1761	EXAMPLE 4:	Moves up the containment tree two levels.
1762		<pre>-&gt; cd ../../</pre>
1763		<pre>/</pre>
1764	EXAMPLE 5:	Changes the Current Default Target to the (absolute) target <code>/system1/cpu1</code> .
1765		<pre>-&gt; cd /system1/cpu1</pre>
1766		<pre>/system1/cpu1</pre>
1767	EXAMPLE 6:	Changes the Current Default Target to <code>/system1/cpu2/temp1</code> (assuming a target named with a UFIT of <code>temp1</code> exists within <code>/system1/cpu2</code> ).
1768		
1769		<pre>-&gt; cd temp1</pre>
1770		<pre>/system1/cpu2/temp1</pre>
1771	EXAMPLE 7:	Moves to the session's Current Default Target's parent and then to <code>cpu1</code> .
1772		<pre>-&gt; cd ../cpu1</pre>
1773		<pre>/system1/cpu1</pre>
1774	EXAMPLE 8:	Returns the error (the target does not resolve) and then returns the current target.
1775		<pre>-&gt; cd cpu1</pre>
1776		No such target found
1777		<pre>/system1/cpu2</pre>
1778	EXAMPLE 9:	Returns the error (the target does not resolve) and then returns the current target.
1779		<pre>-&gt; cd -o format=clpxml /cpu1</pre>
1780		<pre>&lt;?xml version="1.0" encoding="UTF-8"?&gt;</pre>
1781		<pre>&lt;response</pre>
1782		<pre>  xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"</pre>
1783		<pre>  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"</pre>
1784		<pre>  xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0</pre>
1785		<pre>  /dsp0224.xsd smclp_command_response.xsd"&gt;</pre>
1786		<pre>  &lt;command&gt;</pre>
1787		<pre>    &lt;inputline&gt;cd -o format=clpxml /cpu1&lt;/inputline&gt;</pre>
1788		<pre>  &lt;/command&gt;</pre>
1789		<pre>  &lt;cmdstat&gt;</pre>
1790		<pre>    &lt;status&gt;3&lt;/status&gt;</pre>
1791		<pre>    &lt;status_tag&gt;COMMAND EXECUTION FAILED&lt;/status_tag&gt;</pre>
1792		<pre>    &lt;job&gt;</pre>
1793		<pre>      &lt;job_id&gt;243&lt;/job_id&gt;</pre>
1794		<pre>      &lt;joberr&gt;</pre>
1795		<pre>        &lt;errtype&gt;1&lt;/errtype&gt;</pre>
1796		<pre>        &lt;cimstat&gt;6&lt;/cimstat&gt;</pre>
1797		<pre>        &lt;cimstat_desc&gt;CIM_ERR_NOT_FOUND&lt;/cimstat_desc&gt;</pre>
1798		<pre>        &lt;severity&gt;2&lt;/severity&gt;</pre>
1799		<pre>      &lt;/joberr&gt;</pre>
1800		<pre>    &lt;/job&gt;</pre>
1801		<pre>  &lt;/cmdstat&gt;</pre>
1802		<pre>&lt;cd&gt;</pre>
1803		<pre>  &lt;ufip&gt;/system1&lt;/ufip&gt;</pre>
1804		<pre>&lt;/cd&gt;</pre>
1805		<pre>&lt;/response&gt;</pre>

## Server Management Command Line Protocol (SM CLP) Specification

1806 EXAMPLE 10: Changes Current Default Target to /system1/cpu3.

```
1807 -> cd -o format=clpxml /system1/cpu3
1808 <?xml version="1.0" encoding="UTF-8"?>
1809 <response
1810 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
1811 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1812 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
1813 /dsp0224.xsd smclp_command_response.xsd">
1814   <command>
1815     <inputline>cd -o format=clpxml /system1/cpu3</inputline>
1816   </command>
1817   <cmdstat>
1818     <status>0</status>
1819     <status_tag>COMMAND COMPLETED</status_tag>
1820   <job>
1821     <job_id>3</job_id>
1822   </job>
1823 </cmdstat>
1824 <cd>
1825   <ufip>/system1/cpu3</ufip>
1826 </cd>
1827 </response>
```

1828 EXAMPLE 11: Changes Current Default Target to /system1/cpu3.

```
1829 -> cd -o format=keyword /system1/cpu3
1830 commandline=cd -o format=keyword /system1/cpu3
1831 status=0
1832 status_tag=COMMAND COMPLETED
1833 job_id=3
1834 command=cd
1835 ufip=/system1/cpu3
1836 endoutput
```

1837 EXAMPLE 12: Attempts to change Current Default Target to system3, and system3 is currently unresponsive.

```
1838 -> cd -o format=keyword /system3
1839 commandline=cd -o format=keyword /system3
1840 status=3
1841 status_tag=COMMAND EXECUTION FAILED
1842 job_id=3
1843 errtype=2
1844 cimstat=6
1845 severity=2
1846 command=cd
1847 ufip=/system3
1848 endoutput
```

1849 EXAMPLE 13: Bad syntax on command target term.

```
1850 -> cd -output format=keyword //system3
1851 commandline=cd -output format=keyword //system3
1852 status=2
1853 status_tag=COMMAND PROCESSING FAILED
1854 error=246
1855 error_tag=INVALID TARGET
1856 command=cd
1857 ufip=/system3
1858 endoutput
```

## Server Management Command Line Protocol (SM CLP) Specification

1859 EXAMPLE 14: Changes Current Default Target to SESSION.

```
1860 -> cd SESSION  
1861 /map1/settings1/setting5
```

### 1862 6.3 create

1863 The general form of the `create` command is:

```
1864 create [<options>] <target> [<property of new target>=<value>] [<property of new  
1865 target>=<value>]
```

#### 1866 6.3.1 General

1867 For the `create` command, implementations shall support the syntax defined for the `create-cmd` term in  
1868 the CLP grammar defined in Annex A.

1869 The `create` command is used to create new target objects in the target address space of the  
1870 implementation. The `create` command is supported only in certain specific target profiles. This  
1871 command will be supported on any implementation capable of creating new target objects. An example is  
1872 log records. The exact support will be determined by the profiles supported on that implementation.

1873 When creating a new instance, the command target is the object to be created. The class of the object  
1874 being created is determined by the class tag section of the target address passed to the command. If the  
1875 final term of the Resultant Address is a specific UFiT, the implementation shall create an instance with the  
1876 specific UFiT if possible or return an error if creation is not possible. If the Resultant Target terminates in  
1877 a UFST, the path up to and including the penultimate term determines the effective target container where  
1878 the implementation shall create an instance of the class identified by the UFCT specified by the UFST, if  
1879 possible, or return an error if it is not possible.

1880 The `create` command does not support usage without Command Line parameters. When a command  
1881 target term is not included in the Command Line, the implementation shall not execute the command and  
1882 shall return a Command Status of COMMAND PROCESSING ERROR and a Processing Error of  
1883 MISSING REQUIRED TARGET. The `create` command can either be used with property name and  
1884 value pairs or with the `source` option. When property name and value pairs are specified as target  
1885 property terms, the implementation will attempt to assign each named property the associated value. If  
1886 there are additional properties on the instance, the implementation will provide default values. If the  
1887 implementation cannot assign default values to unspecified properties, it will return an error. The required  
1888 properties, default values, and so on are documented in the CLP-to-CIM mapping for the profile that  
1889 defines the target. When the `create` verb is specified without the `source` option, or at least one target  
1890 property term, the implementation will return a Processing Error of COMMAND SYNTAX ERROR. When  
1891 the `source` option and at least one target property term are both specified in a command, the  
1892 implementation will return a Processing Error of COMMAND SYNTAX ERROR. The number of options  
1893 and properties required will vary depending on the command and the target. Specific per-target required  
1894 options and properties are documented in DSP0216.

#### 1895 6.3.2 Valid Targets

1896 This command is supported when it is specified in a target mapping (DSP0216) for a profile that the  
1897 implementation supports. For all targets that do not support the use of the `create` command,  
1898 implementations shall not show the `create` command in a command listing as being available. The  
1899 behavior of this command does change on a per-target basis. Implementations of the `create` command  
1900 will accept an Absolute or a Relative Target Address for the command target term. If the Resultant  
1901 Address is not a UFiP and does not terminate in a UFST, implementations shall return a Command Status  
1902 of COMMAND PROCESSING ERROR and a Processing Error of INVALID TARGET.



1903 **6.3.3 Options**

1904 Implementations of the `create` command support the options specified in **Error! Reference source not**  
 1905 **found..** The use of the `source` option is as follows:

1906     **-source**     The `source` option is used to identify a Managed Element or other data source that will  
 1907                    provide initial values for the instance to be created. Examples include a Managed  
 1908                    Element that will be cloned or settings to use. When this option is supported for use  
 1909                    with a particular target, it is defined in the CLP-to-CIM mapping for that target.

1910 **6.3.4 Output**

1911 This clause details the requirements for output of the `create` verb.

1912 **6.3.4.1 Text Format**

1913 If an object was created, the implementation shall return information about the created object appropriate  
 1914 to the output modifiers selected by the command. If no object was created, the implementation shall  
 1915 indicate this result.

1916 **6.3.4.2 Structured Format**

1917 This clause details requirements for structured output formats for the `create` verb.

1918 **6.3.4.2.1 General**

1919 The returned data shall include any status data in the standard format at the top of the response. The  
 1920 `create` command shall then return a list of the created objects with the keyword "instance". If the  
 1921 implementation cannot create any objects, it should not return any additional data or keywords.

1922 **6.3.4.2.2 XML Output**

1923 The implementation shall return the `create` element in the `response` element in the returned XML  
 1924 document as defined in the Command Response schema in DSP0224.

```

1925 <create>
1926   <instance>
1927     <ufip>User Friendly instance Path identifying target </ufip>
1928     <properties>Properties of the Managed Element.
1929       <property>A property of Managed Element. Each property element is defined
1930       per the xsd.</property>
1931       <property>A property of Managed Element. Each property element is defined
1932       per the xsd.</property>
1933     </properties>
1934   </instance>
1935 </create>
    
```

1936 **6.3.4.2.3 Keyword**

1937 Implementations shall use the following form when returning Command Results for the `create` command  
 1938 in "keyword" format:

```

1939 command=create
1940 begingroup=instance
1941 ufip=User Friendly instance Path of created instance
1942 endgroup
1943 endoutput
    
```

1944 **6.3.5 Examples**

1945 The following examples try to create a log entry in a log.

1946 EXAMPLE 1: Successfully creates a record log.

```
1947 -> create log1/record* event="The dummy test ran" `
1948 probablecause="Running a dummy test" `
1949 recommendedactions="Don't run dummy test" `
1950 severity=unknown source=me
1951 Event = The dummy test ran
1952 RecordID = 75
1953 ProbableCause = Running a dummy test
1954 RecommendedActions = Don't run dummy test
1955 Severity = unknown
1956 Source = me
```

1957 EXAMPLE 2: Fails to create a log record due to missing properties.

```
1958 -> create log1/record*
1959 Create failed--missing required properties.
```

1960 EXAMPLE 3: Fails to create a log record due to missing properties.

```
1961 -> create -output format=clpxml log1/record*
1962 <?xml version="1.0" encoding="UTF-8"?>
1963 <response
1964 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
1965 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1966 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
1967 /dsp0224.xsd
1968 smclp_command_response.xsd">
1969   <command>
1970     <inputline>create -output format=clpxml log1/record*</inputline>
1971   </command>
1972   <cmdstat>
1973     <status>3</status>
1974     <status_tag>COMMAND EXECUTION FAILED</status_tag>
1975     <job>
1976       <job_id>89</job_id>
1977       <joberr>
1978         <errtype>1</errtype>
1979         <errtype_desc>Unknown</errtype_desc>
1980         <cimstat>4</cimstat>
1981         <cimstat_desc>CIM_ERR_INVALID_PARAMETER</cimstat_desc>
1982         <severity>2</severity>
1983       </joberr>
1984     </job>
1985   </cmdstat>
1986   <create></create>
1987 </response>
```

1988 EXAMPLE 4: Fails to create a log record due to missing properties.

```
1989 -> create -output format=keyword log1/record*
1990 commandline=create -output format=keyword log1/record*
1991 status=3
1992 status_tag=COMMAND EXECUTION FAILED
1993 job_id=89
1994 errtype=1
```

## Server Management Command Line Protocol (SM CLP) Specification

```
1995 errtype_desc=unknown
1996 cimstat=4
1997 cimstat_desc=CIM_ERROR_INVALID_PARAMETER
1998 severity=2
1999 command=create
2000 endoutput
```

2001 EXAMPLE 5: Successfully creates a new user account on the MAP.

```
2002 -> create -o format=clpxml /map1/user* userid=someuser
2003 password=somepassword
2004 <?xml version="1.0" encoding="UTF-8"?>
2005 <response
2006 xmlns=http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
2007 xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance"
2008 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
2009 /dsp0224.xsd
2010 smclp_command_response.xsd">
2011   <command>
2012     <inputline>create -o format=clpxml /map1/user* userid=someuser
2013       password=somepassword</inputline>
2014   </command>
2015   <cmdstat>
2016     <status>0</status>
2017     <job>
2018       <job_id>45</job_id>
2019     </job>
2020   </cmdstat>
2021   <create>
2022     <instance>
2023       <ufit ufct="user" instance="4">user4</ufit>
2024       <ufip>/map1/user4</ufip>
2025       <properties>
2026         <property>
2027           <name>userid</name>
2028           <value>
2029             <val>someuser</val>
2030           </value>
2031         </property>
2032         <property>
2033           <name>password</name>
2034           <value>
2035             <val></val>
2036           </value>
2037         </property>
2038       </properties>
2039     </instance>
2040   </create>
2041 </response>
```

## Server Management Command Line Protocol (SM CLP) Specification

2042 EXAMPLE 6: Successfully creates a new user account on the MAP.

```
2043 -> create -o format=keyword /map1/user* userid=someuser  
2044 password=somepassword  
2045 commandline=create -o format=keyword /map1/user userid=someuser  
2046 password=somepassword  
2047 status=0  
2048 job_id=45  
2049 command=create  
2050 begingroup=instance  
2051 ufip=/map1/user4  
2052 endgroup  
2053 endoutput
```

### 2054 6.4 delete

2055 The general form of the `delete` command is:

```
2056 delete [<options>] <target>
```

#### 2057 6.4.1 General

2058 For the `delete` command, implementations shall support the syntax defined for the `delete-cmd` term in  
2059 the CLP grammar defined in Annex A.

2060 The `delete` command is used to remove a target. The `delete` command is supported only in certain  
2061 specific target profiles. This command shall be supported on any implementation capable of deleting  
2062 target objects. The exact support on a MAP will be determined by the profiles supported in that  
2063 implementation.

2064 This command can be used with and without Command Line options. If the Resultant Target terminates in  
2065 a UFsT, the path up to and including the penultimate term determines the target of the command. The  
2066 implementation shall delete all instances of the type specified by the UFcT indicated by the UFsT that are  
2067 immediately contained in the target or return an error.

2068 Even if the target for the `delete` command is such that executing the command will result in deleting the  
2069 Managed Element indicated by the CDT, the implementation shall delete all of the referenced targets in  
2070 the scope, including the target referenced by the CDT. It is possible that the CDT will then refer to a target  
2071 that has been deleted. The user will discover that the CDT is invalid the next time the Resultant Target for  
2072 a command is the same as the value of the CDT.

#### 2073 6.4.2 Valid Targets

2074 This command is supported when it is specified in a target mapping (DSP0216) for a profile that the  
2075 implementation supports. For all targets that do not support the use of the `delete` command,  
2076 implementations shall not show the `delete` command in a command listing as being available. The  
2077 behavior of this command does not change on a per-target basis. Implementations of the `delete`  
2078 command will accept an Absolute or a Relative Target Address for the command target term. If the  
2079 Resultant Address is not a UFiP and does not terminate in a UFsT, implementations shall return a  
2080 Command Status of COMMAND PROCESSING ERROR and a Processing Error of INVALID TARGET.

2081 **6.4.3 Options**

2082 Implementations of the `delete` command will support the options specified in **Error! Reference source**  
 2083 **not found.** The implementation will support other options for the command as specified in the specific  
 2084 target profile.

2085       **-f, -force**       The `force` option allows objects to be deleted, ignoring any policy that might  
 2086                            cause the implementation to normally not execute the command. When this option  
 2087                            is used, the implementation shall execute this deletion if at all possible, without  
 2088                            regard to consequences.

2089 **6.4.4 Output**

2090 This clause details the requirements for output of the `delete` verb.

2091 **6.4.4.1 Text Format**

2092 Implementations shall return Command Result data that includes a list of deleted targets when the  
 2093 `verbose` argument is included with the `output` option, and implementations may specify the list of  
 2094 deleted targets using range notation. If no targets are deleted, the implementation shall indicate that no  
 2095 targets were deleted. For example, the implementation could return the string "No targets deleted" or an  
 2096 appropriate translation.

2097 **6.4.4.2 Structured Format**

2098 This clause details requirements for structured output formats for the `delete` verb.

2099 **6.4.4.2.1 General**

2100 The returned data shall include any status data in the standard format at the top of the response. The  
 2101 `delete` command shall then return a list of the deleted objects.

2102 **6.4.4.2.2 XML Output**

2103 The implementation shall return the `delete` element in the `response` element as defined in the  
 2104 Command Response schema in DSP0224.

```

2105     <delete>
2106         <target>
2107             <instance>
2108                 <ufip>
2109                     UFiP identifying target.
2110                 </ufip>
2111             </instance>
2112             <target>
2113                 Recursive target elements representing Managed Elements contained in
2114                 the initial target.
2115             </target>
2116             .
2117             .
2118             .
2119         </target>
2120         .
2121         .
2122         .
2123     </delete>
    
```

## Server Management Command Line Protocol (SM CLP) Specification

### 2124 6.4.4.2.3 Keyword

2125 Implementations shall use the following form when returning Command Results for the `delete` command  
2126 in "keyword" format:

```
2127     command=delete  
2128     ufip= UFiP of deleted instance  
2129     .  
2130     .  
2131     .  
2132     ufip= UFiP of deleted instance  
2133     endoutput
```

### 2134 6.4.5 Examples

2135 The following examples delete a single log entry on a MAP that supports log deletion. For additional  
2136 examples, see the target profiles that implement this command. Note that in the following examples, if no  
2137 output format is specified, the default output format is in effect. For these examples, the default output  
2138 format is text.

2139 EXAMPLE 1: Deletes the specific log record `record1` contained in `log1`.

```
2140     -> delete log1/record1  
2141     log1/record1 deleted.
```

2142 EXAMPLE 2: Deletes all instances of record contained in `log1` (that is, clears the event log).

```
2143     -> delete -o verbose log1/record*  
2144     record1 deleted.  
2145     record2 deleted.  
2146     record3 deleted.  
2147     record4 deleted.  
2148     record5 deleted.  
2149     5 records successfully deleted.
```

2150 EXAMPLE 3: Deletes all instances of record contained in `log1` (that is, clears the event log). Displays all of the  
2151 results in reverse order.

```
2152     -> delete -o verbose,end,order=reverse log1/record*  
2153     record5 deleted.  
2154     record4 deleted.  
2155     record3 deleted.  
2156     record2 deleted.  
2157     record1 deleted.  
2158     5 records successfully deleted.
```

2159 EXAMPLE 4: Deletes all instances of record contained in `log1`.

```
2160     -> delete -o terse log1/record*  
2161     5 records successfully deleted.
```

2162 EXAMPLE 5: Successfully deletes `record3` in `log1` in `system34`. Requests error-only output, so nothing is  
2163 returned.

```
2164     -> delete -o error log1/record3
```

2165 EXAMPLE 6: Deletes `record3` in `log1` in `system34`.

```
2166     -> delete -o error,format=clpxml /system34/log1/record3  
2167     <?xml version="1.0" encoding="UTF-8"?>  
2168     <response  
2169     xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
```

## Server Management Command Line Protocol (SM CLP) Specification

```
2170 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2171 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
2172 /dsp0224.xsd smclp_command_response.xsd">
2173   <command>
2174     <inputline>delete -o error,format=clpxml
2175 /system34/log1/record3</inputline>
2176   </command>
2177   <cmdstat>
2178     <status>0</status>
2179     <status_tag>COMMAND COMPLETED</status_tag>
2180     <job>
2181       <job_id>45</job_id>
2182     </job>
2183   </cmdstat>
2184   <delete>
2185     <target>
2186       <instance>
2187         <ufit ufct="record" instance="1">record3</ufit>
2188         <ufip>/system34/log1/record3</ufip>
2189       </instance>
2190     </target>
2191   </delete>
2192 </response>
```

2193 EXAMPLE 7: Deletes record3 in log1 in system34.

```
2194 -> delete -o format=keyword log1/record3
2195 commandline=delete -o format=keyword log1/record3
2196 status=0
2197 status_tag=COMMAND COMPLETED
2198 job_id=45
2199 command=delete
2200 ufip=/system34/log1/record3
2201 endoutput
```

2202 EXAMPLE 8: Attempts to delete a record that does not exist.

```
2203 -> delete -o format=clpxml log1/record334
2204 <?xml version="1.0" encoding="UTF-8"?>
2205 <response
2206 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
2207 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2208 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
2209 /dsp0224.xsd smclp_command_response.xsd">
2210   <command>
2211     <inputline>delete -o format=clpxml log1/record334</inputline>
2212   </command>
2213   <cmdstat>
2214     <status>3</status>
2215     <status_tag>COMMAND EXECUTION FAILED</status_tag>
2216     <job>
2217       <job_id>5349</job_id>
2218     <joberr>
2219       <errtype>1</errtype>
2220       <errtype_desc>Other</errtype_desc>
2221     <cimstat>6</cimstat>
2222     <cimstat_desc>CIM_ERR_NOT_FOUND</cimstat_desc>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
2223     <severity>2</severity>
2224     <severity_desc>Low</severity_desc>
2225     <errsource/>
2226     <errsourceform_desc/>
2227   </joberr>
2228 </job>
2229 </cmdstat>
2230 <delete></delete>
2231 </response>
```

2232 EXAMPLE 9: Attempts to delete individual records, but this operation is not supported by this implementation.

```
2233 -> delete -o format=clpxml log3/record334
2234 <?xml version="1.0" encoding="UTF-8"?>
2235 <response
2236 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
2237 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2238 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0/
2239 /dsp0224.xsd smclp_command_response.xsd">
2240   <command>
2241     <inputline>delete -o format=clpxml log3/record334</inputline>
2242   </command>
2243   <cmdstat>
2244     <status>3</status>
2245     <status_tag>COMMAND EXECUTION FAILED</status_tag>
2246   <job>
2247     <job_id>2359</job_id>
2248   <joberr>
2249     <errtype>1</errtype>
2250     <errtype_desc>Other</errtype_desc>
2251     <cimstat>7</cimstat>
2252     <cimstat_desc>CIM_ERR_NOT_SUPPORTED</cimstat_desc>
2253     <severity>2</severity>
2254     <severity_desc>Low</severity_desc>
2255   </joberr>
2256 </job>
2257 </cmdstat>
2258 <delete></delete>
2259 </response>
```

2260 EXAMPLE 10: Attempts to delete individual records, but this operation is not supported by this implementation.

```
2261 -> delete -o format=keyword log3/record334
2262 commandline=delete -o format=keyword log3/record334
2263 status=3
2264 status_tag=COMMAND EXECUTION FAILED
2265 job_id=2359
2266 errtype=1
2267 errtype_desc=Other
2268 cimstat=7
2269 cimstat_desc=CIM_ERR_NOT_SUPPORTED
2270 severity=2
2271 severity_desc=Low
2272 command=delete
2273 endoutput
```



## Server Management Command Line Protocol (SM CLP) Specification

2274 EXAMPLE 11: Deletes all of the users in the current target.

```
2275 -> delete user*
2276 user[1-20] successfully deleted.
```

2277 EXAMPLE 12: Deletes all records in log1 (only four exist).

```
2278 -> delete -o format=clpxml log1/record*
2279 <?xml version="1.0" encoding="UTF-8"?>
2280 <response
2281 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
2282 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2283 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
2284 /dsp0224.xsd smclp_command_response.xsd">
2285   <command>
2286     <inputline>delete -o format=clpxml log1/record* </inputline>
2287   </command>
2288   <cmdstat>
2289     <status>0</status>
2290     <status_tag>COMMAND COMPLETED</status_tag>
2291     <job>
2292       <job_id>45</job_id>
2293     </job>
2294   </cmdstat>
2295   <delete>
2296     <target>
2297       <instance>
2298         <ufit ufct="record" instance="1">record1</ufit>
2299         <ufip>/system34/log1/record1</ufip>
2300       </instance>
2301     </target>
2302     <target>
2303       <instance>
2304         <ufit ufct="record" instance="2">record2</ufit>
2305         <ufip>/system34/log1/record2</ufip>
2306       </instance>
2307     </target>
2308     <target>
2309       <instance>
2310         <ufit ufct="record" instance="3">record3</ufit>
2311         <ufip>/system34/log1/record3</ufip>
2312       </instance>
2313     </target>
2314     <target>
2315       <instance>
2316         <ufit ufct="record" instance="4">record4</ufit>
2317         <ufip>/system34/log1/record4</ufip>
2318       </instance>
2319     </target>
2320   </delete>
2321 </response>
```

2322 EXAMPLE 13: Deletes all records in log1 (only four exist).

```
2323 -> delete -o format=keyword /system34/log1/record*
2324 commandline=delete -o format=keyword /system34/log1/record*
2325 status=0
```

## Server Management Command Line Protocol (SM CLP) Specification

```
2326 status_tag=COMMAND COMPLETED
2327 job_id=45
2328 command=delete
2329 ufip=/system34/log1/record1
2330 ufip=/system34/log1/record2
2331 ufip=/system34/log1/record3
2332 ufip=/system34/log1/record4
2333 endoutput
```

### 2334 6.5 dump

2335 The general form of the `dump` command is:

```
2336 dump -destination <URI> [<options>] [<target>]
```

#### 2337 6.5.1 General

2338 For the `dump` command, implementations shall support the syntax defined for the `dump-cmd` term in the  
2339 CLP grammar defined in Annex A.

2340 The `dump` command is used to take a binary image from an ME and send it to a specific location  
2341 (specified as a URI). This command is supported only on certain specific target profiles. This command  
2342 shall be supported on any implementation that manages binary images. The exact support on a MAP will  
2343 be determined by the profiles supported on that implementation.

2344 If the `destination` option is not supplied by the Client, the implementation shall not execute the  
2345 command and shall return a Command Status of `COMMAND PROCESSING FAILED` and a Processing  
2346 Error of `REQUIRED OPTION MISSING`.

#### 2347 6.5.2 Valid Targets

2348 This command is supported when it is specified in a target mapping (DSP0216) for a profile that the  
2349 implementation supports. For all targets that do not support the use of the `dump` command,  
2350 implementations shall not show the `dump` command in a command listing as being available.  
2351 Implementations of the `dump` command will accept an Absolute or a Relative Target Address for the  
2352 command target term. If the Resultant Address is not a UFiP, implementations shall return a Command  
2353 Status of `COMMAND PROCESSING ERROR` and a Processing Error of `INVALID TARGET`.

#### 2354 6.5.3 Options

2355 Following are valid options for the `dump` command in addition to those specified in **Error! Reference**  
2356 **source not found.**:

```
2357 -destination <URI>
```

2358 The `destination` option tells the implementation the target to which it will transfer the binary image.  
2359 The `destination` option is required on the Command Line every time this verb is executed.

2360 The URI specified can contain a scheme that indicates the explicit service and location to be used to  
2361 capture the dumped data.

#### 2362 6.5.4 Output

2363 This clause details the requirements for output of the `dump` verb.

##### 2364 6.5.4.1 Text Format

2365 Implementations shall include the source and target addresses in the Command Results data and shall  
2366 indicate whether the operation was successful.

## Server Management Command Line Protocol (SM CLP) Specification

2367 EXAMPLE 1: If the command was successful, an implementation could return the following string:

2368 `<target address> transferred to <URI>`

2369 EXAMPLE 2: If the file is not transferred, the implementation could return the following string:

2370 `<target address> not transferred`

### 2371 6.5.4.2 Structured Format

2372 This clause details requirements for structured output formats for the `dump` verb.

#### 2373 6.5.4.2.1 General

2374 The returned data shall include any status data in the standard format at the top of the response. The  
2375 `dump` command shall then return the target address with the keyword `source` and the destination URI  
2376 with the keyword `destination`. If the destination is an address within the MAP address space, the  
2377 implementation shall identify the destination using the keyword `ufip`. If the destination is a URI, the  
2378 implementation shall identify the destination using the keyword `uri`. The Client will need to check the  
2379 Command Status to determine whether the transfer was successful.

#### 2380 6.5.4.2.2 XML Output

2381 The implementation shall return the `dump` element in the `response` element as defined in the Command  
2382 Response schema in DSP0224. A portion of the schema is illustrated below. Note that an implementation  
2383 will return either the `<uri>` or `<ufip>` element as appropriate for the format of the source and destination of  
2384 the command.

```
2385 <dump>  
2386 <source>  
2387   <uri> Full path of source of dump </uri>  
2388 </source>  
2389 <destination>  
2390   <uri> Full path of destination of dump </uri>  
2391 </destination>  
2392 </dump>
```

#### 2393 6.5.4.2.3 Keyword

2394 Implementations shall use the following form when returning Command Results for the `dump` command in  
2395 "keyword" format. If the destination is local to the MAP, the `ufip` keyword shall be used instead of the  
2396 `uri` keyword.

```
2397 command=dump  
2398 begingroup=source  
2399 ufip=UFiP of source  
2400 endgroup  
2401 begingroup=destination  
2402 uri=URI of the destination  
2403 endgroup  
2404 endoutput
```

### 2405 6.5.5 Examples

2406 This clause details examples of the use of the `dump` verb.

2407 EXAMPLE 1: Transfers the binary image of `memory1` to an FTP site.

```
2408 -> dump memory1 -destination ftp://myserver.com/pub/fwimage.img  
2409 memory1 transferred to ftp://myserver.com/pub/fwimage.img.
```

## Server Management Command Line Protocol (SM CLP) Specification

2410 EXAMPLE 2: Attempts to transfer a binary image of `memory1` to an FTP site but the transfer fails because the  
2411 `userid/password` for the destination is invalid.

```
2412 -> dump -destination `
2413 ftp://administrator:passw0rd@myserver.com/private/administrator/
2414 memory.dmp -o format=clpxml memory1
2415 <?xml version="1.0" encoding="UTF-8"?>
2416 <response
2417 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
2418 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2419 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/
2420 1.0.0/dsp0224.xsd
2421 smclp_command_response.xsd">
2422 <command>
2423 <inputline>dump -destination
2424 ftp://administrator:passw0rd@myserver.com/private/
2425 administrator/memory.dmp -o format=clpxml
2426 memory1</inputline>
2427 </command>
2428 <cmdstat>
2429 <status>3</status>
2430 <status_tag>COMMAND EXECUTION FAILED</status_tag>
2431 <job>
2432 <job_id>234324</job_id>
2433 <joberr>
2434 <errtype>1</errtype>
2435 <errtype_desc>Other</errtype_desc>
2436 <cimstat>1</cimstat>
2437 <cimstat_desc>CIM_ERR_FAILED</cimstat_desc>
2438 <severity>2</severity>
2439 <probcause>60</probcause>
2440 <probcause_desc>Login Attempts Failed</probcause_desc>
2441 </joberr>
2442 </job>
2443 </cmdstat>
2444 <dump/>
2445 </response>
```

2446 EXAMPLE 3: Transfer fails because the `userid/password` for the destination is invalid.

```
2447 -> dump -destination `
2448 ftp://administrator:passw0rd@myserver.com/private/administrator/memory.
2449 dmp -o format=keyword memory1
2450 commandline=dump -destination
2451 ftp://administrator:passw0rd@myserver.com/private/administrator/memory.
2452 dmp -o format=keyword memory1
2453 status=3
2454 status_tag=COMMAND EXECUTION FAILED
2455 job_id=234324
2456 errtype=1
2457 errtype_desc=Other
2458 cimstat=1
2459 cimstat_desc=CIM_ERR_FAILED
2460 severity=2
2461 severity_desc=Low
2462 probcause=60
2463 probcause_desc=Login Attempts Failed
2464 command=dump
```

## Server Management Command Line Protocol (SM CLP) Specification

2465 endoutput

2466 EXAMPLE 4: Requests help for the dump command.

```
2467 -> dump -help -o format=clpxml
2468 <?xml version="1.0" encoding="UTF-8"?>
2469 <response
2470 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
2471 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2472 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/
2473 1.0.0/dsp0224.xsd
2474 smclp_command_response.xsd">
2475   <command>
2476     <inputline>dump -help -o format=clpxml</inputline>
2477   </command>
2478   <cmdstat>
2479     <status>0</status>
2480     <job>
2481       <job_id>26210</job_id>
2482     </job>
2483   </cmdstat>
2484   <dump>
2485     <help>
2486       <text>The dump command is used to take a binary image from
2487       an ME and transfer it to another location. This
2488       destination can be within the MAP and specified using a
2489       UFiP. The destination need not be within the MAP. If the
2490       destination is not within the MAP it can be specified
2491       using a URI.</text>
2492     </help>
2493   </dump>
2494 </response>
```

2495 EXAMPLE 5: Requests help for the dump command.

```
2496 -> dump -help
2497 commandline=dump -help
2498 status=0
2499 job_id=26210
2500 command=dump
2501 help=The dump command is used to take a binary image from an ME and
2502 transfer it to another location. This destination can be within the MAP
2503 and specified using a UFiP. The destination need not be within the MAP.
2504 If the destination is not within the MAP it can be specified using a
2505 URI.
2506 endoutput
```

2507 EXAMPLE 6: Transfers memory1.

```
2508 -> dump -destination `
2509 ftp://administrator:passw0rd@myserver.com/private/administrator/memory.
2510 dmp -o format=clpxml memory1
2511 <?xml version="1.0" encoding="UTF-8"?>
2512 <response
2513 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
2514 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2515 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
2516 /dsp0224.xsd smclp_command_response.xsd">
2517   <command>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
2518         <inputline>dump -destination
2519         ftp://administrator:passw0rd@myserver.com/private/
2520         administrator/memory.dmp -o format=clpxml
2521         memory1</inputline>
2522     </command>
2523     <cmdstat>
2524         <status>0</status>
2525         <job>
2526             <job_id>385</job_id>
2527         </job>
2528     </cmdstat>
2529     <dump>
2530         <source>
2531             <ufip>/system1/memory1</ufip>
2532         </source>
2533         <destination>
2534             <uri>ftp://administrator:passw0rd@myserver.com/private/
2535             administrator/memory.dmp</uri>
2536         </destination>
2537     </dump>
2538 </response>
```

2539 EXAMPLE 7: Command is missing the required destination option.

```
2540     -> dump -o format=clpxml memory1
2541     <?xml version="1.0" encoding="UTF-8"?>
2542     <response
2543     xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
2544     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2545     xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
2546     /dsp0224.xsd smclp_command_response.xsd">
2547         <command>
2548             <inputline>dump -o format=clpxml memory1</inputline>
2549         </command>
2550         <cmdstat>
2551             <status>2</status>
2552             <error>251</error>
2553             <error_tag>REQUIRED OPTION MISSING</error_tag>
2554         </cmdstat>
2555         <dump></dump>
2556     </response>
```

2557 EXAMPLE 8: Spawns a job to transfer memory1.

```
2558     -> dump -destination `
2559     ftp://administrator:passw0rd@myserver.com/private/administrator/memory.
2560     dmp -o format=clpxml memory1
2561     <?xml version="1.0" encoding="UTF-8"?>
2562     <response
2563     xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
2564     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2565     xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
2566     /dsp0224.xsd smclp_command_response.xsd">
2567         <command>
2568             <inputline>dump -destination
2569
2570             ftp://administrator:passw0rd@myserver.com/private/administrator/
2571             memory.dmp -o format=clpxml memory1</inputline>
2572         </command>
```

```

2573 <cmdstat>
2574 <status>1</status>
2575 <job>
2576 <job_id>385</job_id>
2577 </job>
2578 </cmdstat>
2579 <dump>
2580 <source>
2581 <ufip>/system1/memory1</ufip>
2582 </source>
2583 <destination>
2584 <uri>ftp://administrator:passw0rd@myserver.com/private/
2585 administrator/memory.dmp</uri>
2586 </destination>
2587 </dump>
2588 </response>

```

2589 EXAMPLE 9: Spawns a job to transfer memory1.

```

2590 -> dump -destination `
2591 ftp://administrator:passw0rd@myserver.com/private/administrator/memory.
2592 dmp -o format=keyword memory1
2593 commandline=dump -destination
2594 ftp://administrator:passw0rd@myserver.com/private/administrator/memory.
2595 dmp memory1
2596 status=1
2597 job_id=385
2598 command=dump
2599 begingroup=source
2600 ufip=/system1/memory1
2601 endgroup
2602 begingroup=destination
2603 uri=ftp://administrator:passw0rd@myserver.com/private/administrator/mem
2604 ory.dmp
2605 endgroup
2606 endoutput

```

## 2607 6.6 exit

2608 The general form of the `exit` command is:

```
2609 exit [<options>]
```

### 2610 6.6.1 General

2611 For the `exit` command, implementations shall support the syntax defined for the `exit-cmd` term in the  
2612 CLP grammar defined in Annex A.

2613 The `exit` command terminates the user's current CLP session. This command shall be supported. When  
2614 this command is received, implementations shall initiate a graceful shutdown of the underlying transport.  
2615 Prior to initiating the shutdown, implementations shall return Command Response data indicating that the  
2616 shutdown has been initiated and should wait for the message to be received by the Client prior to ending  
2617 the session.

## Server Management Command Line Protocol (SM CLP) Specification

### 2618 6.6.2 Valid Targets

2619 The `exit` command has an Implicit Command Target of the Managed Element representing the CLP  
2620 session where the command is issued. If the Command Line includes a command target term, the  
2621 implementation shall not execute the command and shall return a Command Status of COMMAND  
2622 PROCESSING FAILED and a Processing Error of COMMAND SYNTAX ERROR in the Command  
2623 Response data.

### 2624 6.6.3 Options

2625 Implementations of the `exit` command will support the options specified in **Error! Reference source not  
2626 found..**

### 2627 6.6.4 Output

2628 This clause describes requirements for CLP output for the `exit` verb.

#### 2629 6.6.4.1 Text Format

2630 The Command Response data shall include Command Status.

#### 2631 6.6.4.2 Structured Format

2632 This clause details requirements for structured output formats for the `exit` verb.

##### 2633 6.6.4.2.1 General

2634 The returned data shall include any status data in the standard format at the top of the response.

##### 2635 6.6.4.2.2 XML Output

2636 The implementation shall return the `exit` element in the `response` element as defined in the Command  
2637 Response schema in DSP0224.

```
2638 <exit>  
2639 </exit>
```

##### 2640 6.6.4.2.3 Keyword

2641 Implementations shall use the following form when returning Command Results for the `exit` command in  
2642 "keyword" format:

```
2643 command=exit  
2644 endoutput
```

### 2645 6.6.5 Examples

2646 This clause provides examples of the use of the `exit` verb.

2647 EXAMPLE 1: Examines the effect of the `exit` command.

```
2648 -> exit -x  
2649     If run without the examine option, this command will exit the  
2650     current CLP session.
```

2651 EXAMPLE 2: Displays help for the `exit` command.

```
2652 -> exit -help  
2653     The exit command is used to exit a CLP session.
```



2654 EXAMPLE 3: Exits the current session.

```

2655 -> exit -output format=clpxml
2656 <?xml version="1.0" encoding="UTF-8"?>
2657 <response
2658 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
2659 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2660 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
2661 /dsp0224.xsd smclp_command_response.xsd">
2662     <command>
2663         <inputline>exit -output format=clpxml</inputline>
2664     </command>
2665     <cmdstat>
2666         <status>0</status>
2667         <job>
2668             <job_id>2332</job_id>
2669         </job>
2670     </cmdstat>
2671     <exit></exit>
2672 </response>

```

## 2673 6.7 help

2674 The general form of the help command is:

```
2675 help [<options>] [<help topics>]
```

### 2676 6.7.1 General

2677 For the help command, implementations shall support the syntax defined for the help-cmd term in the  
2678 CLP grammar defined in Annex A.

2679 The help command is used to request information related to the use of the CLP. The text that is returned  
2680 by this command can be defined by an OEM as required for its specific market. The help command  
2681 accepts zero or more options. The help command can be specified with zero or more tokens identifying  
2682 topics for which the user is requesting help. Examples of possible tokens that an implementation could  
2683 recognize include verb names, option names, target addresses, UFCts, and target property names. The  
2684 topics recognized and supported by an implementation are implementation specific. If the implementation  
2685 recognizes a token as identifying a topic for which it can provide specific help text, the implementation  
2686 may return help text specific to the topic identified by the token. Implementations of the help command  
2687 shall implement the rules for recognizing and using option terms as specified in 5.2.1.3.3. The help  
2688 command is an exception to the general rules regarding recognizing command target terms and target  
2689 property terms.

### 2690 6.7.2 Valid Targets

2691 The help command is unique in that it does not operate against a target. An implementation may  
2692 recognize a token as a target address and provide help specific to that target.

### 2693 6.7.3 Options

2694 Implementations of the help command will support the following options, in addition to those specified in  
2695 **Error! Reference source not found.** These option arguments may have no effect (return the same data  
2696 as if they were not used) on some implementations.

2697	-output verbose	Implementation should return extensive help text.
2698	-output terse	Implementation should return a short form of help text.

### 2699 6.7.4 Output

2700 This clause states requirements for CLP output for the `help` verb..

#### 2701 6.7.4.1 Text Format

2702 The implementation should return text providing help to the user.

#### 2703 6.7.4.2 Structured Format

2704 This clause details requirements for structured output formats for the `help` verb.

##### 2705 6.7.4.2.1 General

2706 The returned data shall include any status data in the standard format at the top of the response. The  
2707 `help` command shall then return an OEM-defined set of text describing the help for the target or  
2708 command specified on the Command Line. The keyword "helptext" shall be used when returning this text.

##### 2709 6.7.4.2.2 XML Output

2710 The implementation shall return the `help` element in the `response` element as defined in the Command  
2711 Response schema in DSP0224.

```
2712 <help>  
2713   <text>  
2714     :  
2715     Free-form text  
2716     :  
2717   </text>  
2718 </help>
```

##### 2719 6.7.4.2.3 Keyword

2720 Implementations shall use the following form when returning Command Results for the `help` command in  
2721 "keyword" format:

```
2722 command=help  
2723 help=The help text.  
2724 Endoutput
```

### 2725 6.7.5 Examples

2726 This clause provides examples of the use of the `help` verb.

2727 EXAMPLE 1: Displays help for the `log1` target.

```
2728 -> help log1  
2729 log1 is a message log and has records in it.
```

2730 EXAMPLE 2: Uses the `examine` option with the `help` command.

```
2731 -> help -x -o format=clpxml /system1  
2732 <?xml version="1.0" encoding="UTF-8"?>  
2733 <response  
2734 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"  
2735 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
2736 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0/  
2737 /dsp0224.xsd smclp_command_response.xsd">  
2738   <command>  
2739     <inputline>help -x -o format=clpxml /system1</inputline>  
2740   </command>
```

```

2741     <cmdstat>
2742         <status>0</status>
2743     </cmdstat>
2744     <help>
2745         <examine>
2746             <text>If run without the examine option, this command will
2747 return help about "/system1."</text>
2748         </examine>
2749     </help>
2750 </response>

```

2751 EXAMPLE 3: Uses the examine option with the help command.

```

2752 -> help -x -o format=keyword /system1
2753 commandline=help -x -o format=keyword system1
2754 status=0
2755 job_id=989
2756 command=help
2757 examine=If run without the examine option, this command will return
2758 help about "/system1".
2759 endoutput

```

## 2760 6.8 load

2761 The general form of the load command is:

```
2762 load -source <URI> [<options>] [<target>]
```

### 2763 6.8.1 General

2764 For the load command, implementations shall support the syntax defined for the load-cmd term in the  
2765 CLP grammar defined in Annex A.

2766 The load command is used to take a binary image from a specific source location (specified as a URI)  
2767 and place it at the specified target address. The exact behavior of the load command is profile and  
2768 implementation specific. The profile dictates whether the desired action is a simple file transfer or whether  
2769 it includes an implicit installation of the transferred image. In the case of an implicit installation, it is  
2770 implementation dependent whether additional actions are required to complete the installation process.  
2771 This command is supported only on certain specific target profiles. The load command will be supported  
2772 on implementations that manage binary images. The exact support on a MAP will be determined by the  
2773 profiles supported on that implementation.

2774 If the source option is not supplied by the Client, the implementation shall not execute the command  
2775 and shall return a Command Status of COMMAND PROCESSING FAILED and a Processing Error of  
2776 REQUIRED OPTION MISSING.

### 2777 6.8.2 Valid Targets

2778 This command is supported when it is specified in a target mapping (DSP0216) for a profile that the  
2779 implementation supports. For all targets that do not support the use of the load command,  
2780 implementations shall not show the load in a command listing as being available. Implementations of the  
2781 load command will accept an Absolute or a Relative Target Address for the command target term. If the  
2782 Resultant Address is not a UFiP, implementations shall return a Command Status of COMMAND  
2783 PROCESSING ERROR and a Processing Error of INVALID TARGET.

### 2784 6.8.3 Options

2785 Implementations of the `load` command support the following option, in addition to those specified in  
2786 **Error! Reference source not found.:**

2787        `-source <URI>`    This option tells the implementation the target from which it will transfer the  
2788                               binary image.

2789 The URI specified can contain a scheme that indicates the explicit service and location to be used to  
2790 retrieve the binary image.

### 2791 6.8.4 Output

2792 This clause states the requirements for CLP output for the `load` verb.

#### 2793 6.8.4.1 Text Format

2794 The Command Result data shall include the source URI and the target instance address, and shall  
2795 indicate whether the command was successful.

2796 EXAMPLE 1:        If the command was successful, an implementation could return the following string:

2797                    `<URI> transferred to <target address>`

2798 EXAMPLE 2:        If the image is not transferred successfully, the implementation could return the following string:

2799                    `<URI> not transferred`

#### 2800 6.8.4.2 Structured Format

2801 This clause details requirements for structured output formats for the `load` verb.

##### 2802 6.8.4.2.1 General

2803 The returned data shall include any status data in the standard format at the top of the response. The  
2804 `load` command shall then return the target address with the keyword `destination` and the source URI  
2805 with the keyword `source`. If the image is not transferred, the implementation should return only the URI  
2806 address (with the `source` keyword). If the source is an address within the MAP address space, the  
2807 implementation shall identify the source using the keyword `ufip`. If the destination is a URI, the  
2808 implementation shall identify the source using the keyword `uri`.

##### 2809 6.8.4.2.2 XML Output

2810 The implementation shall return the `load` element in the `response` element as defined in the Command  
2811 Response schema in DSP0224. A portion of the schema is illustrated below. Note that an implementation  
2812 will either return the `<uri>` or `<ufip>` element as appropriate for the format of the source and destination of  
2813 the command.

```
2814 <load>
2815   <source>
2816     <uri> Full path of source of load </uri>
2817   </source>
2818   <destination>
2819     <uri> Full path of destination of load </uri>
2820   </destination>
2821 </load>
```

2822 **6.8.4.2.3 Keyword**

2823 Implementations shall use the following form when returning Command Results for the load command in  
2824 "keyword" format:

```
2825     command=load
2826     begingroup=source
2827     uri= URI of the source of the image to load
2828     endgroup
2829     begingroup=destination
2830     ufip= UFiP of the destination for the image
2831     endgroup
2832     endoutput
```

2833 **6.8.5 Examples**

2834 This clause provides examples of the use of the load verb.

2835 EXAMPLE 1: Loads firmware image from FTP site.

```
2836     -> load -source ftp://myserver.com/pub/fwimage.img ` firmwareimage
2837         ftp://myserver.com/pub/firmwareimage.img is transferred to
2838         firmwareimage.
```

2839 EXAMPLE 2: Loads firmware image from an authenticated FTP site.

```
2840     -> load -source ` ftp://administrator:passw0rd@myserver.com/private/ `
2841     administrator/firmware.img -o format=clpxml softwareid1
2842     <?xml version="1.0" encoding="UTF-8"?>
2843     <response
2844     xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
2845     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2846     xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
2847     /dsp0224.xsd smclp_command_response.xsd">
2848         <command>
2849             <inputline>load -source
2850             ftp://administrator:passw0rd@myserver.com/private/
2851             administrator/firmware.img -o format=clpxml
2852             softwareid1</inputline>
2853         </command>
2854         <cmdstat>
2855             <status>0</status>
2856             <job>
2857                 <job_id>385</job_id>
2858             </job>
2859         </cmdstat>
2860         <load>
2861             <source>
2862                 <uri>ftp://administrator:passw0rd@myserver.com/private/
2863                 administrator/firmware.img</uri>
2864             </source>
2865             <destination>
2866                 <ufip>/system1/softwareid1</ufip>
2867             </destination>
2868         </load>
2869     </response>
```

2870 EXAMPLE 3: Loads firmware image from an authenticated FTP site.

```
2871     -> load -source ` ftp://administrator:passw0rd@myserver.com/private `
2872     administrator/firmware.img -o format=keyword softwareid1
```

## Server Management Command Line Protocol (SM CLP) Specification

```
2873      commandline=load -source
2874      ftp://administrator:passwd@myserver.com/private/administrator/
2875      firmware.img -o format=keyword softwareid1
2876      status=0
2877      job_id=385
2878      command=load
2879      begingroup=source
2880      uri=ftp://administrator:passwd@myserver.com/private/administrator/
2881      firmware.img
2882      endgroup
2883      begingroup=destination
2884      ufip=/system1/softwareid1
2885      endgroup
2886      endoutput
```

2887 **EXAMPLE 4:** Fails to load firmware image from an authenticated FTP site due to bad credentials.

```
2888      -> load -source `ftp://administrator:passwd@myserver.com/private/`
2889      administrator/firmware.img -o format=clpxml softwareid1
2890      <?xml version="1.0" encoding="UTF-8"?>
2891      <response
2892      xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
2893      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2894      xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
2895      /dsp0224.xsd
2896      smclp_command_response.xsd">
2897      <command>
2898      <inputline>load -source
2899      ftp://administrator:passwd@myserver.com/private/
2900      administrator/firmware.img -o format=clpxml
2901      softwareid1</inputline>
2902      </command>
2903      <cmdstat>
2904      <status>2</status>
2905      <status_tag>COMMAND EXECUTION FAILED</status_tag>
2906      <job>
2907      <job_id>234324</job_id>
2908      <joberr>
2909      <errtype>1</errtype>
2910      <errtype_desc>Other</errtype_desc>
2911      <cimstat>1</cimstat>
2912      <cimstat_desc>CIM_ERR_FAILED</cimstat_desc>
2913      <severity>2</severity>
2914      <probcause>60</probcause>
2915      <probcause_desc>Login Attempts Failed</probcause_desc>
2916      </joberr>
2917      </job>
2918      </cmdstat>
2919      <dump/>
2920      </response>
```

## 2921 **6.9 reset**

2922 The general form of the reset command is:

```
2923      reset [<options>] [<target>]
```

2924 **6.9.1 General**

2925 For the `reset` command, implementations shall support the syntax defined for the `reset-cmd` term in  
2926 the CLP grammar defined in Annex A.

2927 The `reset` command resets the target's state. This behavior can be modified to take the target to a  
2928 specific state through the use of options.

2929 This command can be used with and without Command Line options.

2930 **6.9.2 Valid Targets**

2931 This command is supported when it is specified in a target mapping (DSP0216) for a profile that the  
2932 implementation supports. For all targets that do not support the use of the `reset` command,  
2933 implementations shall not show the `reset` command in a command listing as being available.

2934 Implementations of the `reset` command will accept an Absolute or a Relative Target Address for the  
2935 command target term. If the Resultant Address is not a UFiP, implementations shall return a Command  
2936 Status of COMMAND PROCESSING ERROR and a Processing Error of INVALID TARGET.

2937 The behavior of state-change commands for each UFcT is defined in DSP0216.

2938 **6.9.3 Options**

2939 Following are valid options for the `reset` command in addition to those specified in **Error! Reference**  
2940 **source not found.:**

2941 `-f, -force` Forces the implementation to reset the object, ignoring any policy that might  
2942 cause the implementation to normally not execute the command. The  
2943 implementation shall execute this reset if at all possible, without regard to  
2944 consequences.

2945 **6.9.4 Output**

2946 This clause states requirements for output for the `reset` verb.

2947 **6.9.4.1 Text Format**

2948 Implementations shall return Command Result data that includes the target address that was reset (if  
2949 any) and the time and date when the reset started. Implementations are free to return the time and date in  
2950 any format that meets their needs. If no targets were reset, the implementation shall indicate this result.

2951 **6.9.4.2 Structured Format**

2952 This clause details requirements for structured output formats for the `reset` verb.

2953 **6.9.4.2.1 General**

2954 Implementations shall include any status data in the standard format at the top of the response. If the  
2955 target was successfully reset, the implementation shall then return the target and the time the reset was  
2956 initiated.

2957 **6.9.4.2.2 XML Output**

2958 The implementation shall return the `reset` element in the `response` element as defined in the  
2959 Command Response schema in DSP0224.

2960 `<reset>`  
2961 `<ufip> Target address the command was invoked against </ufip>`

## Server Management Command Line Protocol (SM CLP) Specification

```
2962     <timestamp> Time reset occurred if completed synchronously, returned in CIM  
2963     datetime format </timestamp>  
2964 </reset>
```

### 6.9.4.2.3 Keyword

2966 Implementations shall use the following form when returning Command Results for the `reset` command  
2967 in "keyword" format:

```
2968     command=reset  
2969     ufip=User Friendly instance path of target of reset  
2970     timestamp=Time reset occurred if completed synchronous to command  
2971     endoutput
```

### 6.9.5 Examples

2973 This clause provides examples of the use of the `reset` verb.

2974 EXAMPLE 1: Resets the operating system on `system1`.

```
2975     -> reset /system1  
2976     /system1 reset at 10:40am 1/1/01.
```

2977 EXAMPLE 2: Fails to reset the operating system on `system1`.

```
2978     -> reset -o format=clpxml /system1/os1  
2979     <?xml version="1.0" encoding="UTF-8"?>  
2980     <response  
2981     xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"  
2982     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
2983     xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0/  
2984     /dsp0224.xsd smclp_command_response.xsd">  
2985         <command>  
2986             <inputline>reset -o format=clpxml /system1/os1</inputline>  
2987         </command>  
2988         <cmdstat>  
2989             <status>3</status>  
2990             <job>  
2991                 <job_id>4824</job_id>  
2992             <joberr>  
2993                 <errtype>2</errtype>  
2994                 <cimstat>1</cimstat>  
2995                 <severity>2</severity>  
2996             </joberr>  
2997             </job>  
2998         </cmdstat>  
2999         <reset>  
3000             <instance>  
3001                 <ufit ufct="os" instance="1">os1</ufit>  
3002                 <ufip>/system1/os1</ufip>  
3003             </instance>  
3004         </reset>  
3005     </response>
```

3006 EXAMPLE 3: Fails to reset the operating system on `system1`.

```
3007     -> reset -o format=keyword /system1/os1  
3008     commandline=reset -o format=keyword /system1/os1
```



```

3009     status=3
3010     job_id=4824
3011     errrtye=2
3012     cimstat=1
3013     severity=2
3014     command=reset
3015     ufip=/system1/os1
3016     endoutput

```

3017 EXAMPLE 4: Resets the operating system on system1.

```

3018     -> reset -w -o format=clpxml /system1/os1
3019     <?xml version="1.0" encoding="UTF-8"?>
3020     <response
3021     xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
3022     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3023     xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
3024     /dsp0224.xsd smclp_command_response.xsd">
3025         <command>
3026             <inputline>reset -w -o format=clpxml /system1/os1</inputline>
3027         </command>
3028         <cmdstat>
3029             <status>0</status>
3030             <job>
3031                 <job_id>92341</job_id>
3032             </job>
3033         </cmdstat>
3034         <reset>
3035             <instance>
3036                 <ufit ufct="os" instance="1">os1</ufit>
3037                 <ufip>/system1/os1</ufip>
3038             </instance>
3039             <timestamp>20050130145904.000000-300</timestamp>
3040         </reset>
3041     </response>

```

3042 EXAMPLE 5: Resets the operating system on system1.

```

3043     -> reset -w -o format=keyword /system1/os1
3044     commandline=reset -w -o format=keyword /system1/os1
3045     status=0
3046     job_id=92341
3047     command=reset
3048     ufip=/system1/os1
3049     timestamp=20050130145904.000000-300
3050     endoutput

```

## 3051 6.10 set

3052 The general form of the set command is:

```

3053     set [<options>] [<target>] <propertyname>=<value>

```

### 3054 6.10.1 General

3055 For the set command, implementations shall support the syntax defined for the set-cmd term in the  
3056 CLP grammar defined in Annex A.

## Server Management Command Line Protocol (SM CLP) Specification

3057 The `set` command is used to set the value of one or more of a target's properties. The command can  
3058 accept a command target term and series of keyword=value pairs which it will try to apply.

3059 The implementation may allow the user to set multiple property values for a single target. The  
3060 implementation may set the property values in the order of properties given on the Command Line.

3061 Implementations shall not allow the user to set properties on multiple targets with a single command.

3062 If an error occurs, the implementation may continue to attempt to set properties. For any property where  
3063 the implementation fails to assign the user-supplied value, the implementation may set the property to a  
3064 default value or the implementation may not change the value of the property at all. It is necessary for the  
3065 user to check the command output or the target itself for the value of each property to determine which  
3066 values were set. Applying properties one command at a time is a deterministic way to determine which  
3067 properties get applied for any given command.

3068 It is possible that changing the value of a property specified by the user will result in the implementation  
3069 changing the value of another property which was not specified by the user, in which case the  
3070 implementation should return both properties and their values in the output.

3071 The `set` command requires Command Line arguments.

### 3072 **6.10.2 Valid Targets**

3073 The `set` command is valid for any target/property pair that is not read-only. For all targets that do not  
3074 support the use of the `set` command, the `set` command shall not show up in a command listing as being  
3075 available. Implementations of the `set` command will accept an Absolute or a Relative Target Address for  
3076 the command target term. If the Resultant Address is not a UFIP, implementations shall return a  
3077 Command Status of COMMAND PROCESSING ERROR and a Processing Error of INVALID TARGET.

### 3078 **6.10.3 Options**

3079 Implementations of the `set` command will support the options specified in **Error! Reference source not  
3080 found.**

### 3081 **6.10.4 Output**

3082 This clause states requirements for output for the `set` verb.

#### 3083 **6.10.4.1 Text Format**

3084 Implementations shall return Command Result data that includes each of the properties that were  
3085 specified in the command and their current values. Note that the current value of a property may be  
3086 different from that requested on the Command Line due to implementation constraints, policies, or vendor  
3087 rules.

#### 3088 **6.10.4.2 Structured Format**

3089 This clause details requirements for structured output formats for the `set` verb.

##### 3090 **6.10.4.2.1 General**

3091 The returned data shall include any status data in the standard format at the top of the response. The `set`  
3092 command shall then return a list of the properties that were set with the property name as the keyword  
3093 and the value to which it was set as the value.

3094 **6.10.4.2.2 XML Output**

3095 The implementation shall return the `set` element in the `response` element as defined in the Command  
3096 Response schema in DSP0224.

```

3097 <set>
3098   <instance>
3099     <ufip>
3100       User Friendly instance Path identifying target
3101     </ufip>
3102     <properties>
3103       <property>
3104         A modified property of Managed Element. Each property element
3105         is defined per the xsd.
3106       </property>
3107     </properties>
3108   </instance>
3109 </set>

```

3110 **6.10.4.2.3 Keyword**

3111 Implementations shall use the following form when returning Command Results for the `set` command in  
3112 "keyword" format:

```

3113 command=set
3114 begingroup=instance
3115 ufip=UFiP of Managed Element targeted by command
3116 begingroup=property
3117 property_name=Property name
3118 property_val=Property value
3119 [property_valstring=String corresponding to property value if value/valuemap]
3120 .
3121   Additional values if property is an array
3122 .
3123 property_val=Property value
3124 [property_valstring=String corresponding to property value if value/valuemap]
3125 endgroup
3126 .
3127   Additional property groups
3128 .
3129 endgroup
3130 endoutput

```

3131 **6.10.5 Examples**

3132 This clause provides examples of the use of the `set` verb.

3133 EXAMPLE 1: Sets the system's name to `sam`.

```

3134 -> set /system1 name=sam
3135 name=sam

```

3136 EXAMPLE 2: Sets password to 12345.

```

3137 -> set /map1/user3 password=12345
3138 password=12345

```

## Server Management Command Line Protocol (SM CLP) Specification

3139 EXAMPLE 3: Successfully sets a new userid and password.

```
3140 -> set /map1/user3 userid=joesmith password=passw0rd
3141      userid=joesmith
3142      password=passw0rd
```

3143 EXAMPLE 4: The password fails. Whether the userid is updated is implementation-specific behavior. This
3144 implementation updated the userid. It is also implementation specific whether the password is
3145 echoed to the screen.

```
3146 -> set /map1/user3 userid=joesmith password=12345
3147      Password 12345 does not meet password rules.
3148      userid=joesmith
3149      password=
```

3150 EXAMPLE 5: The password fails. Whether the userid is updated is implementation-specific behavior. This
3151 implementation did not update the userid. It is also implementation specific whether the password
3152 is echoed to the screen.

```
3153 -> set /map1/user3 userid=joesmith password=12345
3154      Password 12345 does not meet password rules.
3155      userid=olduserid
3156      password=
```

3157 EXAMPLE 6: Sets name to a string that contains spaces.

```
3158 -> set /system1 name="Human Resources Server"
3159      name="Human Resources Server"
```

3160 EXAMPLE 7: Fails to enable load balancing on Ethernet port 2 because teamed NIC is not configured. Notice
3161 that other values already exist for the property that could not be set, and these values are
3162 returned.

```
3163 -> set -o format=clpxml enetport2 enabledcapabilities=loadbalancing
3164 <?xml version="1.0" encoding="UTF-8"?>
3165 <response
3166   xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
3167   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3168   xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0/
3169   /dsp0224.xsd smclp_command_response.xsd">
3170   <command>
3171     <inputline>set -o format=clpxml enetport2
3172     enabledcapabilities=loadbalancing</inputline>
3173   </command>
3174   <cmdstat>
3175     <status>3</status>
3176     <job>
3177       <job_id>4758</job_id>
3178       <joberr>
3179         <errtype>1</errtype>
3180         <cimstat>1</cimstat>
3181         <severity>2</severity>
3182         <probcause>108</probcause>
3183         <probcause_desc>Software Environment Problem</probcause_desc>
3184         <recmdaction>Configure partner NIC prior to
3185         enabling.</recmdaction>
3186       </joberr>
3187     </job>
3188   </cmdstat>
3189   <set>
3190     <instance>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
3191     <ufit ufct="port" instance="2">enetport2</ufit>
3192     <ufip>/system78/enetport2</ufip>
3193     <properties>
3194     <property>
3195     <name>enabledcapabilities</name>
3196     <multivalue>
3197     <value>
3198     <val>3</val><valstring>wakeonlan</valstring>
3199     </value>
3200     <value>
3201     <val>2</val><valstring>alertonlan</valstring>
3202     </value>
3203     </multivalue>
3204     </property>
3205     </properties>
3206     </instance>
3207 </set>
3208 </response>
```

3209 **EXAMPLE 8:** Fails to enable load balancing on Ethernet port 2 because teamed NIC is not configured. Notice  
3210 that other values already exist for the property that could not be set, and these values are  
3211 returned.

```
3212 -> set -o format=keyword enetport2 enabledcapabilities=loadbalancing
3213 commandline=set -o format=keyword enetport2
3214 enabledcapabilities=loadbalancing
3215 status=3
3216 job_id=4758
3217 errtype=1
3218 cimstat=1
3219 severity=2
3220 probcause=108
3221 probcause_desc=Software Environment Problem
3222 recmdaction=Configure partner NIC prior to enabling.
3223 command=set
3224 begingroup=instance
3225 ufip=/system78/enetport2
3226 begingroup=property
3227 property_name=enabledcapabilities
3228 property_val=3
3229 property_valstring=wakeonlan
3230 property_val=2
3231 property_valstring=alertonlan
3232 endgroup
3233 endgroup
3234 endoutput
```

3235 **EXAMPLE 9:** Fails to enable load balancing on Ethernet port 2 because teamed NIC is not configured. Notice  
3236 that this property currently does not have any values assigned.

```
3237 -> set -o format=clpxml enetport2 enabledcapabilities=loadbalancing
3238 <?xml version="1.0" encoding="UTF-8"?>
3239 <response
3240 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
3241 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3242 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0/
3243 /dsp0224.xsd smclp_command_response.xsd">
3244     <command>
3245         <inputline>set -o format=clpxml enetport2
3246             enabledcapabilities=loadbalancing</inputline>
3247     </command>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
3248 <cmdstat>
3249   <status>3</status>
3250   <job>
3251     <job_id>4578</job_id>
3252     <joberr>
3253       <errtype>1</errtype>
3254       <cimstat>1</cimstat>
3255       <severity>2</severity>
3256       <probcause>108</probcause>
3257       <probcause_desc>Software Environment
3258         Problem</probcause_desc>
3259       <recmdaction>Configure partner NIC prior to enabling.
3260         </recmdaction>
3261       <messages>
3262         <message>
3263           <owningentity>OEMxyz</owningentity>
3264           <messageid>23</messageid>
3265           <messagetext>NIC Team {2} could not be configured.
3266           NIC {1} must be configured prior to configuring NIC
3267           {3}
3268           </messagetext>
3269           <messagearg>
3270             <index>1</index>
3271             <value>1</value>
3272           </messagearg>
3273           <messagearg>
3274             <index>2</index>
3275             <value>1</value>
3276           </messagearg>
3277           <messagearg>
3278             <index>3</index>
3279             <value>2</value>
3280           </messagearg>
3281         </message>
3282         <message>
3283           <owningentity>OEMxyz</owningentity>
3284           <messageid>1</messageid>
3285         </message>
3286       </messages>
3287     </joberr>
3288   </job>
3289 </cmdstat>
3290 <set>
3291   <instance>
3292     <ufit ufct="port" instance="2">enetport2</ufit>
3293     <ufip>/system78/enetport2</ufip>
3294     <properties>
3295       <property>
3296         <name>enabledcapabilities</name>
3297         <multivalue></multivalue>
3298       </property>
3299     </properties>
3300   </instance>
3301 </set>
3302 </response>
```

## Server Management Command Line Protocol (SM CLP) Specification

3303 EXAMPLE 10: Fails to enable load balancing on Ethernet port 2 because teamed NIC is not configured. Notice  
3304 that this property currently does not have any values assigned.

```
3305 -> set -o format=keyword enetport2 enabledcapabilities=loadbalancing
3306 commandline=set -o format=keyword enetport2
3307 enabledcapabilities=loadbalancing
3308 status=3
3309 job_id=4578
3310 errtype=1
3311 cimstat=1
3312 severity=2
3313 probcause=108
3314 probcause_desc=Software Environment Problem
3315 beginingroup=message
3316 owningentity=OEMxyz
3317 message_id=23
3318 message=NIC Team {2} could not be configured, NIC {1} must be
3319 configured prior to configuring NIC {3}
3320 message_arg=1
3321 message_arg=1
3322 message_arg=2
3323 endgroup
3324 beginingroup=message
3325 owningentity=OEMxyz
3326 message_id=001
3327 message=Please consult product documentation.message_arg=1
3328 endgroup
3329 recmdaction=Configure partner NIC prior to enabling.
3330 command=set
3331 beginingroup
3332 ufip=/system78/enetport2
3333 beginingroup
3334 property_name=enabledcapabilities
3335 endgroup
3336 endoutput
```

3337 EXAMPLE 11: Successfully enables failover and WakeOnLAN support on Ethernet port 2. Notice that multiple  
3338 values are assigned.

```
3339 -> set -o format=clpxml enetport2
3340 enabledcapabilities=failover,wakeonlan
3341 <?xml version="1.0" encoding="UTF-8"?><response
3342 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
3343 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3344 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
3345 /dsp0224.xsd smclp_command_response.xsd">
3346   <command>
3347     <inputline>set -o format=clpxml /system78/enetport2
3348     enabledcapabilities=failover,wakeonlan</inputline>
3349   </command>
3350   <cmdstat>
3351     <status>0</status>
3352     <job>
3353       <job_id>7623</job_id>
3354     </job>
3355   </cmdstat>
3356   <set>
3357     <instance>
3358       <ufit instance="2" ufct="port">enetport2</ufit>
3359       <ufip>/system78/enetport2</ufip>
3360     <properties>
3361       <property>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
3362         <name>enabledcapabilities</name>
3363         <multivalue>
3364             <value>
3365                 <val>4</val>
3366                 <valstring>failover</valstring>
3367             </value>
3368             <value>
3369                 <val>3</val>
3370                 <valstring>wakeonlan</valstring>
3371             </value>
3372         </multivalue>
3373     </property>
3374 </properties>
3375 </instance>
3376 </set>
3377 </response>
```

3378 EXAMPLE 12: Successfully enables failover and WakeOnLAN support on Ethernet port 2. Notice that multiple  
3379 values are assigned.

```
3380 -> set -o format=keyword enetport2
3381 enabledcapabilities=failover,wakeonlan
3382 commandline=set /system78/enetport2
3383 enabledcapabilities=failover,wakeonlan
3384 status=0
3385 job_id=7632
3386 command=set
3387 begingroup=instance
3388 ufip=/system78/enetport2
3389 begingroup=property
3390 property_name=enabledcapabilities
3391 property_val=4
3392 property_valstring=failover
3393 property_val=3
3394 property_valstring=wakeonlan
3395 endgroup
3396 endgroup
3397 endoutput
```

3398 EXAMPLE 13: Sets the default output format to XML.

```
3399 -> set SESSION outputformat=clpxml
3400 /map1/sessions1/setting5
3401     Default output format set to XML.
```

3402 EXAMPLE 14: Changes the CDT by using the back door method.

```
3403 -> set SESSION cdt=/system5/cpu3
3404 /map1/sessions1/setting5
3405     Current Default Target is /system5/cpu3.
```

3406 EXAMPLE 15: Sets the OEM property OEMxyzpropertyA.

```
3407 -> set /system1 OEMxyzpropertyA=somevalue_
3408 /system1
3409     OEMxyzpropertyA equals somevalue.
```

3410 EXAMPLE 16: Sets a property. The vendor returns data outside the specification.

```
3411 -> set -o format=keyword /system1 PrimaryOwnerName=TheOwner
3412 commandline=set -o format=keyword /system1 PrimaryOwnerName=TheOwner
3413 status=0
```



## Server Management Command Line Protocol (SM CLP) Specification

```
3414     job_id=7632
3415     command=set
3416     begingroup=instance
3417     ufip=/system1
3418     begingroup=property
3419     property_name=PrimaryOwnerName
3420     property_val=TheOwner
3421     endgroup
3422     endgroup
3423     oemxyz_message=The Contact information may need to be updated.
3424     endoutput
```

3425 EXAMPLE 17: Enables software2 for memory1. The association is explicitly identified.

```
3426     -> set
3427     /system1/memory1=>ElementSoftwareIdentity=>/system1/swid1/software2
3428     IsCurrent=true
3429     software2 is now active for memory1
```

3430 EXAMPLE 18: Fails to change which software2 is current on memory1 because multiple instances of ElementSoftwareIdentity reference memory1.

```
3432     -> set /system1/memory1=>ElementSoftwareIdentity IsCurrent=true
3433     Failed to set property "IsCurrent". Both references to an Association
3434     are required when it is the target of a set command.
```

3435 EXAMPLE 19: Enables software2. Specifies both references.

```
3436     -> set
3437     /system1/swid1/software2=>ElementSoftwareIdentity=>/system1/memory1
3438     IsCurrent=true
3439     software2 is now active for memory1.
```

3440 EXAMPLE 20: Attempts to suspend a currently running job.

```
3441     -> set /map1/jobqueue1/job23 jobstate=suspend
3442     Failed to suspend job23. The targeted job does not support
3443     suspension.
```

3444 EXAMPLE 21: Stops traffic over nic1.

```
3445     -> set /system4/nic1 enabledstate=quiesce
3446     Traffic over nic1 has been quiesced.
```

3447 EXAMPLE 22: Enables SSHv1 for a session.

```
3448     -> set /system1/sshprotoendpt1 enabledsshversions[1]=sshv1
3449     /system1/sshprotoendpt1
3450     enabledsshversions[0] = SSHv2
3451     enabledsshversions[1] = SSHv1
```

### 3452 6.11 show

3453 The general form of the show command is:

```
3454     show [<options>] [<target>] [<properties>] [<propertyname>== <propertyvalue>]
```

#### 3455 6.11.1 General

3456 For the show command, implementations shall support the syntax defined for the show-cmd term in the  
3457 CLP grammar defined in Annex A.

## Server Management Command Line Protocol (SM CLP) Specification

3458 The `show` command is used to display information about Managed Elements or Associations. It can be  
3459 used to view information about a single Managed Element, a tree of Managed Elements, or Managed  
3460 Elements matching a property value filter. When executed against a Resultant Address that ends in a  
3461 UFIT without any other options, the `show` command will display information about the single instance  
3462 identified by the Resultant Address. When executed against a Resultant Address that ends in a UFsT, the  
3463 `show` command will display information about contained instances of the type specified by the UFcT  
3464 specified in the UFsT. When used with the `level` option, the `show` command can be used to view a tree  
3465 of Managed Elements.

3466 The Command Results for the `show` command is determined in the following order:

- 3467 1) Determine the Resultant Target for the command.
- 3468 2) Select Managed Element and Associations for which results will be returned from the  
3469 containment hierarchy below the Resultant Target based on the value of the `level` option.
- 3470 3) Retrieve results for the selected Managed Elements and Associations.
- 3471 4) If the command target term terminated in a UFsT, apply the UFcT as a filter against the  
3472 Managed Elements.
- 3473 5) If one or more property target terms that include the equivalence operator were specified, filter  
3474 Managed Elements and Associations based on the property/value.
- 3475 6) If one or more target property terms that are property names were specified, filter the results to  
3476 include only Managed Elements and Associations that have the properties, and for each  
3477 Managed Element and Association, filter the results to include only the specified properties.
- 3478 7) If the `display` option was specified, apply its arguments to filter the results.

3479 If the Resultant Address terminates in a UFsT, the path up to and including the penultimate term of the  
3480 Resultant Address determines the Resultant Target of the command. If the Resultant Address contains a  
3481 single term that is a UFsT, the Resultant Target of the command will be the Managed Element that is the  
3482 root of the address space. The implementation shall return each instance that is contained within the  
3483 Resultant Target of the type specified by the UFcT specified in the UFsT. The implementation shall  
3484 search the containment hierarchy below the Resultant Target for instances of the type specified by the  
3485 UFcT specified in the UFsT to the depth specified by the `level` option. The implementation shall return  
3486 information about the contained instances such that the information returned conforms with any  
3487 restrictions or expansions specified by other options to the command. It is possible that zero instances  
3488 are contained and thus there will not be any instances for which to return information. This is not an error  
3489 and will be handled by implementations returning an appropriate representation of an empty result set.

3490 The `show` command can be specified with target property terms. Each target property term will either be a  
3491 property name or contain a property name, equivalence operator, and property value. When specified, the  
3492 target property terms affect the results returned by the `show` command. The `show` command can be used  
3493 with target property terms as follows:

- 3494 • without target property terms
- 3495 • with target property terms that are property names
- 3496 • with target property terms that contain the equivalence operator
- 3497 • with a combination of target property terms that are just property names and others that contain  
3498 the equivalence operator

3499 When the `show` command is used with one or more target property terms that are property names, the  
3500 implementation shall restrict the results shown to include only Managed Elements or Associations that  
3501 have the specified property and restrict the results for the Managed Element or Association to include  
3502 only properties where the property name was specified as a target property term.

3503 Implementations shall accept at least one target property term that contains the equivalence operator with  
3504 the `show` command to use as a filter on the Managed Elements or Associations for which results are  
3505 returned. Implementations may accept more than one target property term that contains an equivalence  
3506 operator with the `show` command to use as a filter on the Managed Elements or Associations for which  
3507 results will be returned. When an implementation accepts exactly one occurrence of a target property  
3508 term that includes the equivalence operator and more than one occurrence of a target property term that  
3509 includes the equivalence operator is specified on the Command Line, the implementation shall return a  
3510 Command Status of `COMMAND PROCESSING FAILED` and a Processing Error of `FUNCTION NOT`  
3511 `SUPPORTED`. For each target property term that includes the equivalence operator specified with the  
3512 `show` command, the implementation shall restrict the Managed Elements or Associations for which  
3513 results are shown to include only those Managed Elements or Associations that have a property with the  
3514 specified name and the value of the property is equivalent to the value specified in the target property  
3515 term.

3516 The `display` option can be used with the `show` command to control the information returned for an  
3517 instance. Using the `properties` argument to the `display` option, with the `show` command a user can  
3518 effectively search the address space (or a branch) for instances having a certain property or a  
3519 property/value pair. When the `show` command and `display` option are used in this fashion, it effectively  
3520 provides a query function. This `display` option can also be used to restrict the Managed Element  
3521 instances and Association instances for which information will be returned by using the `targets` and  
3522 `associations` arguments respectively. For more information on using the `display` option, see **Error!**  
3523 **Reference source not found.**

3524 If the `show` command is specified without the `display` option, implementations will use the session  
3525 default value for the `display` option. Use of the `level` option is supported only when the Resultant  
3526 Address of the command is a UFiP. Regardless of whether the implementation supports the `level`  
3527 option, when the `show` command is specified without the `level` option, the implementation shall behave  
3528 as if the `level` option was specified with an argument of `'1'`. Note that this has no effect when the  
3529 Resultant Target is an Association Class or Association instance.

3530 The command can be used with and without Command Line options.

### 3531 6.11.2 Valid Targets

3532 All targets and target/property combinations are valid for this command, and its behavior generally does  
3533 not change based on target or property. In specific profiles, the behavior can be slightly different as  
3534 defined in that profile. Implementations of the `show` command will accept an Absolute or a Relative  
3535 Target Address for the command target term.

### 3536 6.11.3 Options

3537 Following are valid options for the `show` command in addition to those specified in **Error! Reference**  
3538 **source not found.:**

3539 `-l, -level <value>`

3540 Controls the target depth level for the containment hierarchy retrieval. The default  
3541 for `<value>` is `"1"` (that is, "the current target only"). Other values can retrieve  
3542 "current target plus 'n-1' levels deep". To retrieve all levels recursively, the  
3543 argument value `all` can be used with the `level` option.

3544 `-d, -display <arg values>`

3545 Selects the category of information that is displayed about a target ME. Valid  
3546 option argument values for this option include `associations`, `targets`,  
3547 `properties`, `verbs`, and `all`. OEMs may also add values using the `OEM_`  
3548 namespace as defined in the OEM extensions clause of this document. When

## Server Management Command Line Protocol (SM CLP) Specification

3549 this option is not specified, the `show` command behaves as if this option was  
3550 included with an argument of `all`.

3551 `-a, -all` Instructs the implementation to return all data element types subject to any  
3552 filtering of categories by the `display` option.

3553 Table 13 lists each type of data element that is returned by the `show` command and indicates whether the  
3554 type requires the `all` option to be specified in the Command Line in order to be included in the  
3555 Command Results. A value of "yes" in a cell indicates that elements of the corresponding data element  
3556 type will not be returned unless the `all` option is included with the `show` command. For each data  
3557 element type listed in Table 13 for which the column labeled "-all Required" includes a value of "yes",  
3558 implementations shall return elements of the specified data element type if and only if the `all` option is  
3559 specified. For each data element type listed in Table 13 for which the column labeled "-all Required" is  
3560 blank, implementations shall return elements of the specified data element type irrespective of whether  
3561 the `all` option is specified.

3562 **Table 13 – Data Element Types and `all` Option**

Data Element Type	Corresponding Display Argument	-all Required
Required properties	properties	
Core properties	properties	yes
OEM properties	properties	yes
SM CLP verbs	verbs	
OEM verbs	verbs	yes
Addressing associations	associations	
Non-addressing associations	associations	
SM CLP targets	targets	
OEM targets	targets	yes

3563 An implementation shall include a verb supported by the implementation in the list of verbs for a Managed  
3564 Element if a command that includes the verb will complete successfully when the Resultant Target of the  
3565 command is the Managed Element. An implementation shall not include a verb in the list of verbs for a  
3566 Managed Element if a command that includes the verb will not successfully complete when the Resultant  
3567 Target of the command is the Managed Element.

### 3568 **6.11.4 Output**

3569 This clause states requirements for CLP output for the `show` verb.

#### 3570 **6.11.4.1 Text Format**

3571 When contained targets are returned, the Command Results output shall include a list of contained  
3572 targets. If the Command Results contain multiple targets, the implementation shall return results such that  
3573 the target containment hierarchy is unambiguous. If properties are returned, the implementation shall  
3574 return results such that the target to which the properties belong is unambiguous. If the command  
3575 resulted in no data, the implementation should not return any data.

#### 3576 **6.11.4.2 Structured Format**

3577 This clause details requirements for structured output formats for the `show` verb.

##### 3578 **6.11.4.2.1 General**

3579 The returned data shall include any status data in the standard format at the top of the response.

3580 **6.11.4.2.2 XML Output**

3581 The XML document fragment indicates the general form of XML-encoded Command Results for the `show`  
 3582 command. It is possible that multiple instances of the `<target>` element will be returned. Target  
 3583 containment hierarchy is explicitly indicated through `<target>` element nesting. The implementation shall  
 3584 return the `show` element in the `response` element as defined in the Command Response schema in  
 3585 DSP0224.

```

3586 <show>
3587   <target>
3588     <instance>
3589       <ufip>
3590         User Friendly instance Path identifying target
3591       </ufip>
3592       <properties>Properties of the Managed Element.
3593         <property>A property of Managed Element. Each property element is
3594           defined per the xsd.</property>
3595         <property>A property of Managed Element. Each property element is
3596           defined per the xsd.</property>
3597       </properties>
3598       <associations>
3599         <association>
3600           <ufct>Association class name</ufct>
3601           <ufip>User Friendly instance Path of other referenced Managed
3602             Element.</ufip>
3603         </association>
3604       </associations>
3605       <verbs>
3606         <standardverbs> CLP term separator delimited list of CLP verbs
3607       </standardverbs>
3608         <oemverbs>CLP term separator delimited list of OEM verbs</oemverbs>
3609       </verbs>
3610     </instance>
3611     <target>Recursive target elements representing Managed Elements contained
3612       in initial target.</target>
3613   </target>
3614 </show>

```

3615 **6.11.4.2.3 Keyword**

3616 It is possible that results for multiple Managed Elements will be returned for the `show` command. Each  
 3617 Managed Element is returned as a block starting with:

```
3618 begingroup=instance
```

3619 The target containment hierarchy is not indicated in the Command Results structure. Instead, the  
 3620 `begingroup=instance` keyword/value pair identifies that a listing of UFiPs of contained Managed  
 3621 Elements will follow if this target contains other targets.

3622 Implementations shall use the following form when returning Command Results for the `show` command in  
 3623 "keyword" format:

```

3624 command=show
3625 begingroup=instance
3626 ufip=UFiP of Managed Element results are for
3627 begingroup=property
3628 property_name=property name
3629 property_val=property value

```

## Server Management Command Line Protocol (SM CLP) Specification

```
3630     endgroup
3631     .
3632     Additional property groups
3633     .
3634     begingroup=targets
3635     ufip=UFiP of contained target
3636     .
3637     Additional contained targets
3638     .
3639     endgroup
3640     begingroup=association
3641     ufct=UFcT of association
3642     ufip=UFiP on other end of association
3643     begingroup=property
3644     property_name=Name of Property of association
3645     property_val=value of property of association
3646     endgroup
3647     .
3648     Additional properties of the association
3649     .
3650     endgroup
3651     .
3652     Additional associations referencing this Managed Element
3653     .
3654     begingroup=verbs
3655     verb=Command applicable to this Managed Element
3656     .
3657     Additional commands applicable to this Managed Element
3658     .
3659     endgroup
3660     endgroup
3661     .
3662     Additional Managed Elements that are returned as results
3663     .
3664     endoutput
```

### 3665 6.11.5 Examples

3666 This clause provides examples of the use of the show verb.

3667 EXAMPLE 1: Shows all of the targets in the root of the address space.

```
3668     -> show -display targets /
3669     /
3670     Targets:
3671     map1
3672     system1
3673     system2
3674     hw1
```

3675 EXAMPLE 2: Shows the commands that apply to the root of the address space.

```
3676     -> show -display verbs /
3677     show
3678     cd
```

3679 EXAMPLE 3: Shows the value of the name property on system1.

```
3680     -> show -display properties=name /system1
3681     name = deadweight
```

## Server Management Command Line Protocol (SM CLP) Specification

3682 EXAMPLE 4: Displays the physical containment hierarchy for chassis1.

```
3683 -> show -display targets -level all -o format=clpxml /hw1/chassis1
3684 <?xml version="1.0" encoding="UTF-8"?>
3685 <response
3686 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
3687 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3688 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
3689 /dsp0224.xsd smclp_command_response.xsd">
3690   <command>
3691     <inputline>show -display targets -level all -o format=clpxml
3692       /hw1/chassis1</inputline>
3693   </command>
3694   <cmdstat>
3695     <status>0</status>
3696     <status_tag>COMMAND COMPLETED</status_tag>
3697     <job>
3698       <job_id>8734</job_id>
3699     </job>
3700   </cmdstat>
3701   <show>
3702     <target>
3703       <instance>
3704         <ufit ufct="chassis" instance="1">chassis1</ufit>
3705         <ufip>/chassis1</ufip>
3706       </instance>
3707     <target>
3708       <instance>
3709         <ufit ufct="board" instance="1">board1</ufit>
3710         <ufip>/chassis1/board1</ufip>
3711       </instance>
3712     <target>
3713       <instance>
3714         <ufit ufct="card" instance="1">card1</ufit>
3715         <ufip>/chassis1/board1/card1</ufip>
3716       </instance>
3717     <target>
3718       <instance>
3719         <ufit ufct="chip" instance="1">chip1</ufit>
3720         <ufip>/chassis1/board1/card1/chip1</ufip>
3721       </instance>
3722     </target>
3723     <target>
3724       <instance>
3725         <ufit ufct="chip" instance="2">chip2</ufit>
3726         <ufip>/chassis1/board1/card1/chip2</ufip>
3727       </instance>
3728     </target>
3729   </target>
3730 </target>
3731 <target>
3732   <instance>
3733     <ufit ufct="powerpkg" instance="1">powerpkg1</ufit>
3734     <ufip>/chassis1/powerpkg1</ufip>
3735   </instance>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
3736     </target>
3737     <target>
3738         <instance>
3739             <ufit ufct="powerpkg" instance="2">powerpkg2</ufit>
3740             <ufip>/chassis1/powerpkg2</ufip>
3741         </instance>
3742     </target>
3743     <target>
3744         <instance>
3745             <ufit ufct="fanpkg" instance="1">fanpkg1</ufit>
3746             <ufip>/chassis1/fanpkg1</ufip>
3747         </instance>
3748     </target>
3749     <target>
3750         <instance>
3751             <ufit ufct="fanpkg" instance="2">fanpkg2</ufit>
3752             <ufip>/chassis1/fanpkg2</ufip>
3753         </instance>
3754     </target>
3755     <target>
3756         <instance>
3757             <ufit ufct="fanpkg" instance="3">fanpkg3</ufit>
3758             <ufip>/chassis1/fanpkg3</ufip>
3759         </instance>
3760     </target>
3761     <target>
3762         <instance>
3763             <ufit ufct="fanpkg" instance="4">fanpkg4</ufit>
3764             <ufip>/chassis1/fanpkg4</ufip>
3765         </instance>
3766     </target>
3767 </target>
3768 </show>
3769 </response>
```

3770 **EXAMPLE 5:** Displays the associations that reference the target.

```
3771 -> show -display associations -o format=clpxml /system1/powersup3
3772 <?xml version="1.0" encoding="UTF-8"?>
3773 <response
3774 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
3775 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3776 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0/
3777 /dsp0224.xsd smclp_command_response.xsd">
3778     <command>
3779         <inputline>show -display associations -o format=clpxml
3780             /system1/powersup3</inputline>
3781     </command>
3782     <cmdstat>
3783         <status>0</status>
3784         <status_tag>COMMAND COMPLETED</status_tag>
3785         <job>
3786             <job_id>129</job_id>
3787         </job>
3788     </cmdstat>
3789 </show>
```



## Server Management Command Line Protocol (SM CLP) Specification

```
3790 <target>
3791   <instance>
3792     <ufit ufct="powersup" instance="3">powersup3</ufit>
3793     <ufip>/system1/powersup3</ufip>
3794     <associations>
3795       <association>
3796         <ufct>suppliespower</ufct>
3797         <reference>
3798           <name>dependent</name>
3799           <instance>
3800             <ufit ufct="system"
3801               instance="1">system1</ufit>
3802             <ufip>/system1</ufip>
3803           </instance>
3804         </reference>
3805         <reference>
3806           <name>antecedent</name>
3807           <instance>
3808             <ufit ufct="powersup"
3809               instance="3">powersup3</ufit>
3810             <ufip>/system1/powersup3</ufip>
3811           </instance>
3812         </reference>
3813       </association>
3814       <association>
3815         <ufct>realizes</ufct>
3816         <reference>
3817           <name>antecedent</name>
3818           <instance>
3819             <ufit ufct="powerpkg"
3820               instance="2">powerpkg2</ufit>
3821             <ufip>/chassis23/powerpkg2</ufip>
3822           </instance>
3823         </reference>
3824         <reference>
3825           <name>dependent</name>
3826           <instance>
3827             <ufit ufct="powersup"
3828               instance="3">powersup3</ufit>
3829             <ufip>/system1/powersup3</ufip>
3830           </instance>
3831         </reference>
3832       </association>
3833       <association>
3834         <ufct>SystemDevice</ufct>
3835         <reference>
3836           <name>partcomponent</name>
3837           <instance>
3838             <ufit ufct="powersup"
3839               instance="3">powersup3</ufit>
3840             <ufip>/system1/powersup3</ufip>
3841           </instance>
3842         </reference>
3843         <reference>
3844           <name>groupcomponent</name>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
3845         <instance>
3846             <ufit ufct="system"
3847                 instance="1">system1</ufit>
3848             <ufip>/system1</ufip>
3849         </instance>
3850     </reference>
3851 </association>
3852 </associations>
3853 </instance>
3854 </target>
3855 </show>
3856 </response>
```

3857 **EXAMPLE 6:** Displays the SuppliesPower and Realizes associations that reference the target.

```
3858 -> show -display associations=(SuppliesPower,Realizes) -o format=clpxml
3859 /system1/powersup3
3860 <?xml version="1.0" encoding="UTF-8"?>
3861 <response
3862     xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
3863     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3864     xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
3865     /dsp0224.xsd
3866     smclp_command_response.xsd">
3867     <command>
3868         <inputline>show -display associations=(suppliespower,realizes)
3869             /system1/powersup3</inputline>
3870     </command>
3871     <cmdstat>
3872         <status>0</status>
3873         <status_tag>COMMAND COMPLETED</status_tag>
3874         <job>
3875             <job_id>129</job_id>
3876         </job>
3877     </cmdstat>
3878     <show>
3879         <target>
3880             <instance>
3881                 <ufit ufct="powersup" instance="3">powersup3</ufit>
3882                 <ufip>/system1/powersup3</ufip>
3883                 <associations>
3884                     <association>
3885                         <ufct>suppliespower</ufct>
3886                         <reference>
3887                             <name>dependent</name>
3888                             <instance>
3889                                 <ufit ufct="system"
3890                                     instance="1">system1</ufit>
3891                                 <ufip>/system1</ufip>
3892                             </instance>
3893                         </reference>
3894                         <reference>
3895                             <name>antecedent</name>
3896                             <instance>
3897                                 <ufit ufct="powersup"
3898                                     instance="3">powersup3</ufit>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
3899         <ufip>/system1/powersup3</ufip>
3900     </instance>
3901     </reference>
3902 </association>
3903 <association>
3904     <ufct>realizes</ufct>
3905     <reference>
3906         <name>antecedent</name>
3907         <instance>
3908             <ufit ufct="powerpkg"
3909                 instance="2">powerpkg2</ufit>
3910             <ufip>/chassis23/powerpkg2</ufip>
3911         </instance>
3912     </reference>
3913 </reference>
3914     <name>dependent</name>
3915     <instance>
3916         <ufit ufct="powersup"
3917             instance="3">powersup3</ufit>
3918         <ufip>/system1/powersup3</ufip>
3919     </instance>
3920 </reference>
3921 </association>
3922 </associations>
3923 </instance>
3924 </target>
3925 </show>
3926 </response>
```

3927 **EXAMPLE 7:** Views the status of the spawned job above. In this example, the spawned job failed to run to  
3928 completion.

```
3929 -> show -o format=clpxml -d properties /map1/job1/job385
3930 <?xml version="1.0" encoding="UTF-8"?>
3931 <response
3932 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
3933 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3934 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0/
3935 dsp0224.xsd
3936 smclp_command_response.xsd">
3937     <command>
3938         <inputline>show -o format=clpxml /map1/job1/job385</inputline>
3939     </command>
3940     <cmdstat>
3941         <status>0</status>
3942         <job>
3943             <job_id>892</job_id>
3944         </job>
3945     </cmdstat>
3946     <show>
3947         <target>
3948             <instance>
3949                 <ufit ufct="job" instance="385">job385</ufit>
3950                 <ufip>/map1/jobqueue1/job385</ufip>
3951             <properties>
3952                 <property>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
3953         <name>JobState</name>
3954         <value>
3955             <val>10</val>
3956             <valstring>Exception</valstring>
3957         </value>
3958     </property>
3959     <property>
3960         <name>TimeOfLastStateChange</name>
3961         <value><val>20050130145904.000000-
3962             300</val></value>
3963         <type>datetime</type>
3964     </property>
3965     <property>
3966         <name>TimeBeforeRemoval</name>
3967         <value><val>00000000000500.000000:000</val>
3968     </value>
3969     </property>
3970     <property>
3971         <name>ElapsedTime</name>
3972         <value><val>00000000000034.000000:000</val>
3973     </value>
3974     </property>
3975     <property>
3976         <name>StartTime</name>
3977         <value><val>20050130145830.000000-300</val>
3978     </value>
3979     </property>
3980     <property>
3981         <name>TimeSubmitted</name>
3982         <value><val>20050130145830.000000-300</val>
3983     </value>
3984     </property>
3985     <property>
3986         <name>TimeBeforeRemoval</name>
3987         <value><val>00000000000500.000000:000</val>
3988     </value>
3989     </property>
3990     <property>
3991         <name>name</name>
3992         <value>
3993             <val>dump -destination
3994             ftp://administrator:passw0rd@myserver.com/
3995             private/administrator/memory.dmp memory1
3996             </val>
3997         </value>
3998     </property>
3999     </properties>
4000 </instance>
4001 </target>
4002 </show>
4003 </response>
```

## Server Management Command Line Protocol (SM CLP) Specification

4004 EXAMPLE 8: Queries the value of the PrimaryOwnerName and ResetCapability properties by using `-display`  
4005 `properties=(ResetCapability,PrimaryOwnerName)` as a filter on these results.

```
4006 -> show -display ` properties=(ResetCapability,PrimaryOwnerName)
4007 /system1
4008     /system1
4009         properties
4010             ResetCapability = Disabled (3)
4011             PrimaryOwnerName = Some guy
```

4012 EXAMPLE 9: Queries the value of the PrimaryOwnerName and ResetCapability properties using  
4013 `-display properties=(ResetCapability,PrimaryOwnerName)` as a filter on these  
4014 results.

```
4015 -> show -display ` properties=(ResetCapability,PrimaryOwnerName) -o
4016 format=clpxml /system1
4017 <?xml version="1.0" encoding="UTF-8"?>
4018 <response
4019 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
4020 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4021 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
4022 /dsp0224.xsd smclp_command_response.xsd">
4023     <command>
4024         <inputline>show -display
4025             properties=(ResetCapability,PrimaryOwnerName) -o
4026                 format=clpxml /system1 </inputline>
4027     </command>
4028     <cmdstat>
4029         <status>0</status>
4030     </cmdstat>
4031     <show>
4032         <target>
4033             <instance>
4034                 <ufit ufct="system" instance="1">system1</ufit>
4035                 <ufip>/system1</ufip>
4036                 <properties>
4037                     <property>
4038                         <name>ResetCapability</name>
4039                         <value>
4040                             <val>3</val>
4041                             <valstring>Disabled</valstring>
4042                         </value>
4043                     </property>
4044                     <property>
4045                         <name>Dedicated</name>
4046                         <multivalue>
4047                             <value>
4048                                 <val>4</val>
4049                                 <valstring>Router</valstring>
4050                             </value>
4051                             <value>
4052                                 <val>3</val>
4053                                 <valstring>Switch</valstring>
4054                             </value>
4055                         </multivalue>
4056                     </property>
4057                 </properties>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
4058         </instance>
4059     </target>
4060 </show>
4061 </response>
```

4062 EXAMPLE 10: Displays all of the commands applicable to `system1`.

```
4063 -> show -display verbs -all -o format=clpxml /system1
4064 <?xml version="1.0" encoding="UTF-8"?>
4065 <response
4066 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
4067 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4068 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
4069 /dsp0224.xsd smclp_command_response.xsd">
4070     <command>
4071         <inputline>show -display verbs -all -o format=clpxml
4072         /system1</inputline>
4073     </command>
4074     <cmdstat>
4075         <status>0</status>
4076         <job>
4077             <job_id>2342</job_id>
4078         </job>
4079     </cmdstat>
4080     <show>
4081         <target>
4082             <instance>
4083                 <ufit ufct="system" instance="1">system1</ufit>
4084                 <ufip>/system1</ufip>
4085                 <verbs>
4086                     <standardverbs>show start stop set reset
4087                     help</standardverbs>
4088                     <oemverbs>OEMxyzDoSomething</oemverbs>
4089                 </verbs>
4090             </instance>
4091         </target>
4092     </show>
4093 </response>
```

4094 EXAMPLE 11: Displays information about `system1` and everything immediately contained within it.

```
4095 -> show -d all -level 2 -o format=clpxml /system1
4096 <?xml version="1.0" encoding="UTF-8"?>
4097 <response
4098 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
4099 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4100 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
4101 /dsp0224.xsd smclp_command_response.xsd">
4102     <command>
4103         <inputline>show -d all -level 1 -o format=clpxml
4104         /system1</inputline>
4105     </command>
4106     <cmdstat>
4107         <status>0</status>
4108         <status_tag>COMMAND COMPLETED</status_tag>
4109         <job>
4110             <job_id>29345</job_id>
4111         </job>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
4112 </cmdstat>
4113 <show>
4114   <target>
4115     <instance>
4116       <ufit ufct="system" instance="1">system1</ufit>
4117       <ufip>/system1</ufip>
4118       <properties>
4119         <property>
4120           <name>NameFormat</name>
4121           <value>
4122             <val>IP</val>
4123           </value>
4124         </property>
4125         <property>
4126           <name>ResetCapability</name>
4127           <value>
4128             <val>5</val>
4129           </value>
4130         </property>
4131         <property>
4132           <name>IdentifyingDescriptions</name>
4133           <multivalue>
4134             <value>
4135               <val>mac address</val>
4136             </value>
4137             <value>
4138               <val>mac address</val>
4139             </value>
4140           </multivalue>
4141         </property>
4142         <property>
4143           <name>OtherIdentifyingInfo</name>
4144           <multivalue>
4145             <value>
4146               <val>0023342312</val>
4147             </value>
4148             <value>
4149               <val>0023342313</val>
4150             </value>
4151           </multivalue>
4152         </property>
4153         <property>
4154           <name>Dedicated</name>
4155           <multivalue>
4156             <value>
4157               <val>4</val>
4158               <valstring>Router</valstring>
4159             </value>
4160             <value>
4161               <val>5</val>
4162               <valstring>Switch</valstring>
4163             </value>
4164           </multivalue>
4165         </property>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
4166     </properties>
4167     <associations>
4168         <association>
4169             <ufct>ComputerSystemPackage</ufct>
4170             <reference>
4171                 <name>antecedent</name>
4172                 <instance>
4173                     <ufit ufct="chassis"
4174                         instance="3">chassis3</ufit>
4175                     <ufip>/hw1/chassis3</ufip>
4176                 </instance>
4177             </reference>
4178             <reference>
4179                 <name>dependent</name>
4180                 <instance>
4181                     <ufit ufct="system"
4182                         instance="1">system1</ufit>
4183                     <ufip>/system1</ufip>
4184                 </instance>
4185             </reference>
4186         </properties>
4187         <property>
4188             <name>PlatformGUID</name>
4189             <value>
4190                 <val>00992365293059103762850194833920
4191                 </val>
4192             </value>
4193         </property>
4194     </properties>
4195 </association>
4196 <association>
4197     <ufct>SuppliesPower</ufct>
4198     <reference>
4199         <name>antecedent</name>
4200         <instance>
4201             <ufit ufct="powersupply"
4202                 instance="1">powersupply1</ufit>
4203             <ufip>/system1/powersupply1</ufip>
4204         </instance>
4205     </reference>
4206     <reference>
4207         <name>dependent</name>
4208         <instance>
4209             <ufit ufct="cpu" instance="1">cpul</ufit>
4210             <ufip>/system1</ufip>
4211         </instance>
4212     </reference>
4213 </association>
4214 <association>
4215     <ufct>SuppliesPower</ufct>
4216     <reference>
4217         <name>antecedent</name>
4218         <instance>
4219             <ufit ufct="powersupply"
4220                 instance="2">powersupply2</ufit>
```



## Server Management Command Line Protocol (SM CLP) Specification

```
4221         <ufip>/system1/powersupply2</ufip>
4222     </instance>
4223 </reference>
4224 <reference>
4225     <name>dependent</name>
4226     <instance>
4227         <ufit ufct="cpu" instance="1">cpul</ufit>
4228         <ufip>/system1/cpul</ufip>
4229     </instance>
4230 </reference>
4231 </association>
4232 <association>
4233     <ufct>AssociatedCooling</ufct>
4234     <reference>
4235         <name>antecedent</name>
4236         <instance>
4237             <ufit ufct="fan" instance="1">fan1</ufit>
4238             <ufip>/system1/fan1</ufip>
4239         </instance>
4240     </reference>
4241     <reference>
4242         <name>dependent</name>
4243         <instance>
4244             <ufit ufct="system"
4245                 instance="1">system1</ufit>
4246             <ufip>/system1</ufip>
4247         </instance>
4248     </reference>
4249 </association>
4250 <association>
4251     <ufct>AssociatedCooling</ufct>
4252     <reference>
4253         <name>antecedent</name>
4254         <instance>
4255             <ufit ufct="system"
4256                 instance="1">system1</ufit>
4257             <ufip>/system1/fan2</ufip>
4258         </instance>
4259     </reference>
4260     <reference>
4261         <name>dependent</name>
4262         <instance>
4263             <ufit ufct="system"
4264                 instance="1">system1</ufit>
4265             <ufip>/system1</ufip>
4266         </instance>
4267     </reference>
4268 </association>
4269 <association>
4270     <ufct>Realizes</ufct>
4271     <reference>
4272         <name>antecedent</name>
4273         <instance>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
4274         <ufit ufct="ppowersupply"  
4275             instance="1">ppowersupply1</ufit>  
4276         <ufip>/hw1/chassis3/ppowersupply1</ufip>  
4277     </instance>  
4278 </reference>  
4279 <reference>  
4280     <name>dependent</name>  
4281     <instance>  
4282         <ufit ufct="powersupply"  
4283             instance="1">powersupply1</ufit>  
4284         <ufip>/system1/powersupply1</ufip>  
4285     </instance>  
4286 </reference>  
4287 </association>  
4288 <association>  
4289     <ufct>realizes</ufct>  
4290     <reference>  
4291         <name>dependent</name>  
4292         <instance>  
4293             <ufit ufct="powersupply"  
4294                 instance="2">powersupply2</ufit>  
4295             <ufip>/system1/powersupply2</ufip>  
4296         </instance>  
4297     </reference>  
4298     <reference>  
4299         <name>antecedent</name>  
4300         <instance>  
4301             <ufit ufct="ppowersupply"  
4302                 instance="2">ppowersupply2</ufit>  
4303             <ufip>/hw1/chassis3/ppowersupply2</ufip>  
4304         </instance>  
4305     </reference>  
4306 </association>  
4307 <association>  
4308     <ufct>Realizes</ufct>  
4309     <reference>  
4310         <name>antecedent</name>  
4311         <instance>  
4312             <ufit ufct="ppowersupply"  
4313                 instance="2">ppowersupply2</ufit>  
4314             <ufip>/hw1/chassis3/ppowersupply2</ufip>  
4315         </instance>  
4316     </reference>  
4317     <reference>  
4318         <name>dependent</name>  
4319         <instance>  
4320             <ufit ufct="powersupply"  
4321                 instance="2">powersupply2</ufit>  
4322             <ufip>/system1/powersupply2</ufip>  
4323         </instance>  
4324     </reference>  
4325 </association>  
4326 </associations>  
4327 <verbs>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
4329         <standardverbs>show set stop start
4330         reset</standardverbs>
4331     </verbs>
4332 </instance>
4333 <target>
4334     <instance>
4335         <ufit ufct="cpu" instance="1">cpul</ufit>
4336         <ufip>/system1/cpul</ufip>
4337         <properties>
4338             <property>
4339                 <name>Family</name>
4340                 <value>
4341                     <val>1</val>
4342                 </value>
4343             </property>
4344             <property>
4345                 <name>OtherFamilyDescription</name>
4346                 <value>
4347                     <val>SuperSlow100</val>
4348                 </value>
4349             </property>
4350             <property>
4351                 <name>MaxClockSpeed</name>
4352                 <value>
4353                     <val>33</val>
4354                 </value>
4355                 <units>Megahertz</units>
4356             </property>
4357             <property>
4358                 <name>CPUStatus</name>
4359                 <value>
4360                     <val>1</val>
4361                 </value>
4362             </property>
4363         </properties>
4364     <verbs>
4365         <standardverbs>show</standardverbs>
4366     </verbs>
4367 </instance>
4368 </target>
4369 <target>
4370     <instance>
4371         <ufit ufct="powersup"
4372         instance="1">powersup1</ufit>
4373         <ufip>/system1/powersup1</ufip>
4374         <properties>
4375             <property>
4376                 <name>TotalOutputPower</name>
4377                 <value>
4378                     <val>1200000</val>
4379                 </value>
4380                 <units>milliwatts</units>
4381             </property>
4382         </properties>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
4383         <associations>
4384             <association>
4385                 <ufct>SuppliesPower</ufct>
4386                 <reference>
4387                     <name>dependent</name>
4388                     <instance>
4389                         <ufit ufct="system"
4390                             instance="1">system1</ufit>
4391                         <ufip>/system1</ufip>
4392                     </instance>
4393                 </reference>
4394                 <reference>
4395                     <name>antecedent</name>
4396                     <instance>
4397                         <ufit ufct="powersup"
4398                             instance="1">powersup1</ufit>
4399                         <ufip>/system1/powersup1</ufip>
4400                     </instance>
4401                 </reference>
4402             </association>
4403             <association>
4404                 <ufct>Realizes</ufct>
4405                 <reference>
4406                     <name>antecedent</name>
4407                     <instance>
4408                         <ufit ufct="powerpkg"
4409                             instance="1">powerpkg1</ufit>
4410                         <ufip>/hw1/chassis3/powerpkg1</ufip>
4411                     </instance>
4412                 </reference>
4413                 <reference>
4414                     <name>dependent</name>
4415                     <instance>
4416                         <ufit ufct="powersup"
4417                             instance="1">powersup1</ufit>
4418                         <ufip>/system1/powersup1</ufip>
4419                     </instance>
4420                 </reference>
4421             </association>
4422         </associations>
4423         <verbs>
4424             <standardverbs>show</standardverbs>
4425         </verbs>
4426     </instance>
4427 </target>
4428 <target>
4429     <instance>
4430         <ufit ufct="powersup"
4431             instance="2">powersup2</ufit>
4432         <ufip>/system1/powersup2</ufip>
4433     <properties>
4434         <property>
4435             <name>TotalOutputPower</name>
4436             <value>
4437                 <val>1200000</val>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
4438         </value>
4439         <units>milliwatts</units>
4440     </property>
4441 </properties>
4442 <associations>
4443     <association>
4444         <ufct>SuppliesPower</ufct>
4445         <reference>
4446             <name>dependent</name>
4447             <instance>
4448                 <ufit ufct="system"
4449                     instance="1">system1</ufit>
4450                 <ufip>/system1</ufip>
4451             </instance>
4452         </reference>
4453         <reference>
4454             <name>antecedent</name>
4455             <instance>
4456                 <ufit ufct="powersup"
4457                     instance="2">powersup2</ufit>
4458                 <ufip>/system1/powersup2</ufip>
4459             </instance>
4460         </reference>
4461     </association>
4462     <association>
4463         <ufct>Realizes</ufct>
4464         <reference>
4465             <name>antecedent</name>
4466             <instance>
4467                 <ufit ufct="powerpkg"
4468                     instance="2">powerpkg2</ufit>
4469                 <ufip>/hw1/chassis3/powerpkg2</ufip>
4470             </instance>
4471         </reference>
4472         <reference>
4473             <name>dependent</name>
4474             <instance>
4475                 <ufit ufct="powersup"
4476                     instance="2">powersup2</ufit>
4477                 <ufip>/system1/powersup2</ufip>
4478             </instance>
4479         </reference>
4480     </association>
4481 </associations>
4482 <verbs>
4483     <standardverbs>show</standardverbs>
4484 </verbs>
4485 </instance>
4486 </target>
4487 <target>
4488     <instance>
4489         <ufit ufct="fan" instance="1">fan1</ufit>
4490         <ufip>/system1/fan1</ufip>
4491     </properties>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
4492         <property>
4493             <name>VariableSpeed</name>
4494             <value>
4495                 <val>true</val>
4496             </value>
4497         </property>
4498         <property>
4499             <name>DesiredSpeed</name>
4500             <value>
4501                 <val>3600</val>
4502             </value>
4503             <units>Revolutions per minute</units>
4504         </property>
4505         <property>
4506             <name>ActiveCooling</name>
4507             <value>
4508                 <val>True</val>
4509             </value>
4510         </property>
4511     </properties>
4512     <associations>
4513         <association>
4514             <ufct>AssociatedCooling</ufct>
4515             <reference>
4516                 <name>dependent</name>
4517                 <instance>
4518                     <ufit ufct="system"
4519                         instance="1">system1</ufit>
4520                     <ufip>/system1</ufip>
4521                 </instance>
4522             </reference>
4523             <reference>
4524                 <name>antecedent</name>
4525                 <instance>
4526                     <ufit ufct="fan"
4527                         instance="1">fan1</ufit>
4528                     <ufip>/system1/fan1</ufip>
4529                 </instance>
4530             </reference>
4531         </association>
4532     </associations>
4533     <verbs>
4534         <standardverbs>show set</standardverbs>
4535     </verbs>
4536 </instance>
4537 </target>
4538 <target>
4539     <instance>
4540         <ufit ufct="fan" instance="2">fan2</ufit>
4541         <ufip>/system1/fan2</ufip>
4542     <properties>
4543         <property>
4544             <name>VariableSpeed</name>
4545             <value>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
4546         <val>true</val>
4547     </value>
4548 </property>
4549 <property>
4550     <name>DesiredSpeed</name>
4551     <value>
4552         <val>3600</val>
4553     </value>
4554     <units>Revolutions per minute</units>
4555 </property>
4556 <property>
4557     <name>ActiveCooling</name>
4558     <value>
4559         <val>True</val>
4560     </value>
4561 </property>
4562 </properties>
4563 <associations>
4564     <association>
4565         <ufct>AssociatedCooling</ufct>
4566         <reference>
4567             <name>dependent</name>
4568             <instance>
4569                 <ufit ufct="system"
4570                     instance="1">system1</ufit>
4571                 <ufip>/system1</ufip>
4572             </instance>
4573         </reference>
4574         <reference>
4575             <name>antecedent</name>
4576             <instance>
4577                 <ufit ufct="fan"
4578                     instance="2">fan2</ufit>
4579                 <ufip>/system1/fan2</ufip>
4580             </instance>
4581         </reference>
4582     </association>
4583 </associations>
4584 <verbs>
4585     <standardverbs>show set</standardverbs>
4586 </verbs>
4587 </instance>
4588 </target>
4589 </target>
4590 </show>
4591 </response>
```

4592 **EXAMPLE 12:** Displays information about `system1` and everything immediately contained within it.

```
4593 -> show -d all -level 2 -o format=keyword /system1
4594 commandline=show -d all -level 2 -o format=keyword /system1
4595 status=0
4596 status_tag=COMMAND COMPLETED
4597 job_id=29345
4598 command=show
4599 begingroup=instance
4600 ufip=/system1
4601 begingroup=property
4602 property_name=NameFormat
4603 property_val=IP
```

## Server Management Command Line Protocol (SM CLP) Specification

```
4604     endgroup
4605     begingroup=property
4606     property_name=ResetCapability
4607     property_val=5
4608     endgroup
4609     begingroup=property
4610     property_name=IdentifyingDescriptions
4611     property_val=mac address
4612     property_val=mac address
4613     endgroup
4614     begingroup=property
4615     property_name=OtherIdentifyingInfo
4616     property_val=0023342312
4617     property_val=0023342313
4618     endgroup
4619     begingroup=property
4620     property_name=Dedicated
4621     property_val=4
4622     property_valstring=Router
4623     property_val=5
4624     property_valstring=Switch
4625     endgroup
4626     begingroup=targets
4627     ufip=powersup1
4628     ufip=powersup2
4629     ufip=fan1
4630     ufip=fan2
4631     endgroup
4632     begingroup=association
4633     ufct=ComputerSystemPackage
4634     ufip=/hw1/chassis3
4635     begingroup=property
4636     property_name=PlatformGUID
4637     property_val=00992365293059103762850194833920
4638     endgroup
4639     endgroup
4640     begingroup=association
4641     ufct=SuppliesPower
4642     ufip=/system1/powersup1
4643     endgroup
4644     begingroup=association
4645     ufct=SuppliesPower
4646     ufip=/system1/powersup2
4647     endgroup
4648     begingroup=association
4649     ufct=AssociatedCooling
4650     ufip=/system1/fan1
4651     endgroup
4652     begingroup=association
4653     ufct=AssociatedCooling
4654     ufip=/system1/fan2
4655     endgroup
4656     begingroup=verbs
4657     verb=reset
4658     verb=start
4659     verb=stop
4660     verb=set
4661     verb=show
4662     endgroup
4663     endgroup
4664     begingroup=instance
4665     ufip=/system1/cpu1
4666     begingroup=property
4667     property_name=Family
```



## Server Management Command Line Protocol (SM CLP) Specification

```
4668     property_val=1
4669     endgroup
4670     begingroup=instance
4671     property_name=OtherFamilyDescription
4672     property_val=SuperSlow100
4673     endgroup
4674     begingroup=instance
4675     property_name=MaxClockSpeed
4676     property_val=33
4677     units=Megahertz
4678     endgroup
4679     begingroup=instance
4680     property_name=CPUStatus
4681     property_val=1
4682     endgroup
4683     begingroup=verbs
4684     verb=show
4685
4686     endgroup
4687     endgroup
4688     begingroup=instance
4689     ufip=/system1/powersup1
4690     begingroup=property
4691     property_name=TotalOutputPower
4692     property_val=1200000
4693     units=milliwatts
4694     begingroup=association
4695     ufct=Realizes
4696     ufip=/hw1/chassis3/powerpkg1
4697     endgroup
4698     begingroup=association
4699     ufct=SuppliesPower
4700     ufip=/system1
4701     endgroup
4702     begingroup=verbs
4703     verb=show
4704     endgroup
4705     begingroup=instance
4706     ufip=/system1/powersup2
4707     begingroup=property
4708     property_name=TotalOutputPower
4709     property_val=1200000
4710     units=milliwatts
4711     endgroup
4712     begingroup=association
4713     ufct=Realizes
4714     ufip=/hw1/chassis3/powerpkg2
4715     endgroup
4716     begingroup=association
4717     ufct=SuppliesPower
4718     ufip=/system1
4719     endgroup
4720     begingroup=verbs
4721     verb=show
4722     endgroup
4723     begingroup=instance
4724     ufip=/system1/fan1
4725     begingroup=property
4726     property_name=DesiredSpeed
4727     property_val=3600
4728     units=Revolutions per Minutes
4729     endgroup
4730     begingroup=property
4731     property_name=VariableSpeed
```

## Server Management Command Line Protocol (SM CLP) Specification

```
4732     property_val=true
4733     property_name=ActiveCooling
4734     property_val=true
4735     endgroup
4736     beingroup=association
4737     ufct=AssociatedCooling
4738     ufip=/system1
4739     endgroup
4740     beingroup=verbs
4741     verb=show
4742     verb=set
4743     endgroup
4744     beingroup=instance
4745     ufip=/system1/fan2
4746     beingroup=property
4747     property_name=DesiredSpeed
4748     property_val=3600
4749     units=Revolutions per Minutes
4750     endgroup
4751     beingroup=property
4752     property_name=VariableSpeed
4753     property_val=true
4754     endgroup
4755     beingroup=property
4756     property_name=ActiveCooling
4757     property_val=true
4758     endgroup
4759     beingroup=association
4760     ufct=AssociatedCooling
4761     ufip=/system1
4762     endgroup
4763     beingroup=verbs
4764     verb=show
4765     verb=set
4766     endgroup
4767     endoutput
```

4768 **EXAMPLE 13:** Shows the OperationalStatus of components in system1.

```
4769     -> show -display properties=(OperationalStatus,Name) ~ -level 2
4770     /system1
4771     /system1
4772         OperationalStatus is OK
4773         Name is webserver1
4774     /system1/cpu1
4775         OperationalStatus is OK
4776         Name is main processor 1
4777     /system1/cpu2
4778         OperationalStatus is OK
4779         Name is main processor 2
4780     /system1/cpu3
4781         OperationalStatus is OK
4782         Name is main processor 3
4783     /system1/cpu4
4784         OperationalStatus is Degraded
4785         Name is main processor 4
4786     /system1/powersup1
4787         OperationalStatus is Degraded
4788         Name is AC power 1
4789     /system1/powersup2
4790         OperationalStatus is OK
```

## Server Management Command Line Protocol (SM CLP) Specification

4791                   Name is AC power 2

4792   EXAMPLE 14:   Shows all of the CPUs that have a bad operational status.

4793                   -> show -display properties=(Name,OperationalStatus) ` /system1/cpu\*  
4794                   OperationalStatus==Degraded  
4795                    /system1/cpu4  
4796                    Name is main processor 4  
4797                    OperationalStatus is Degraded

4798   EXAMPLE 15:   Shows all of the components that have a bad operational status.

4799                   -> show -level 2 -display properties=(OperationalStatus) ` /system1  
4800                   (OperationalStatus==Degraded)  
4801                    /system1/cpu4  
4802                    OperationalStatus is Degraded  
4803                    /system1/powersup1  
4804                    OperationalStatus is Degraded

4805   EXAMPLE 16:   Shows all of the components that are named "main processor 4".

4806                   -> show -level 2 -display properties=(Name) /system1 ` Name=="Main  
4807                   processor 4"  
4808                    /system1/cpu4  
4809                    Name is main processor 4

4810   EXAMPLE 17:   Tries to filter using two property values.

4811                   -> show -level 2 -display properties=(Name) /system1 'Name=="Main  
4812                   processor 4",OperationalStatus==Degraded  
4813                    Filtering with multiple properties is not supported.

4814   EXAMPLE 18:   Shows just the operational status of the processors in system1.

4815                   -> show /system1/cpu\* OperationalStatus  
4816                    /system1/cpu1  
4817                    OperationalStatus = Degraded  
4818                    /system1/cpu2  
4819                    OperationalStatus = Ok

4820   EXAMPLE 19:   Assuming the same system1 as the preceding example, shows the ElementName of all  
4821                   degraded processors in system1.

4822                   -> show /system1/cpu\* ElementName OperationalStatus==Degraded  
4823                    /system1/cpu1  
4824                    ElementName = "processor one"

4825   EXAMPLE 20:   Assuming the same system1 as the preceding example, shows the OperationalStatus of all  
4826                   degraded processors in system1.

4827                   -> show /system1/cpu\* OperationalStatus ' OperationalStatus==Degraded  
4828                    /system1/cpu1  
4829                    OperationalStatus = Degraded

## Server Management Command Line Protocol (SM CLP) Specification

4830 EXAMPLE 21: Shows the values for options that can have default values.

```
4831 -> show -display properties -o format=clpxml SESSION
4832 <?xml version="1.0" encoding="UTF-8"?>
4833 <response
4834 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
4835 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4836 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
4837 /dsp0224.xsd smclp_command_response.xsd">
4838   <command>
4839     <inputline>show -display properties -o format=clpxml
4840     SESSION</inputline>
4841   </command>
4842   <cmdstat>
4843     <status>0</status>
4844   </cmdstat>
4845   <show>
4846     <target>
4847       <instance>
4848         <ufit ufct="clpendpt" instance="5">clpendpt5</ufit>
4849         <ufip>/map1/sessions1/clpendpt5</ufip>
4850       <properties>
4851         <property>
4852           <name>cdt</name>
4853           <value>
4854             <val>/system1/cpu4</val>
4855           </value>
4856         </property>
4857         <property>
4858           <name>outputformat</name>
4859           <value>
4860             <val>4</val><valstring>xml</valstring>
4861           </value>
4862         </property>
4863         <property>
4864           <name>outputverbosity</name>
4865           <value>
4866             <val>terse</val>
4867             <valstring>1</valstring>
4868           </value>
4869         </property>
4870         <property>
4871           <name>outputlanguage</name>
4872           <value>
4873             <val>eng</val>
4874           </value>
4875         </property>
4876         <property>
4877           <name>outputposition</name>
4878           <value>
4879             <val>1</val>
4880             <valstring>begin</valstring>
4881           </value>
4882         </property>
4883       </properties>
```

```

4884         <name>outputorder</name>
4885         <value>
4886             <val>1</val>
4887             <valstring>default</valstring>
4888         </value>
4889     </property>
4890 <property>
4891     <name>outputcount</name>
4892     <value>
4893         <val>0</val>
4894     </value>
4895 </property>
4896 <property>
4897     <name>keep</name>
4898     <value>
4899         <val>300</val>
4900     </value>
4901 </property>
4902 <property>
4903     <name>wait</name>
4904     <value>
4905         <val>>false</val>
4906     </value>
4907 </property>
4908 </properties>
4909 </instance>
4910 </target>
4911 </show>
4912 </response>

```

4913 EXAMPLE 22: Shows information about the processors in system1.

```

4914 -> show -display properties=(Name),targets=cpu Name=="Main processor 4"
4915     /system1/cpu4
4916     Name is main processor 4

```

## 4917 6.12 start

4918 The general form of the `start` command is:

```
4919 start [<options>] [<target>]
```

### 4920 6.12.1 General

4921 For the `start` command, implementations shall support the syntax defined for the `start-cmd` term in the CLP grammar defined in Annex A.

4923 The `start` command starts the target. If the target is already started, an error might or might not be returned. The precise behavior is profile specific.

4925 This command can be used with and without Command Line options.

### 4926 6.12.2 Valid Targets

4927 This command is supported when it is specified in a target mapping (DSP0216) for a profile that the implementation supports. For all targets that do not support the use of the `start` command,

## Server Management Command Line Protocol (SM CLP) Specification

4929 implementations shall not show the `start` command in a command listing as being available.  
4930 Implementations of the `start` command will accept an Absolute or a Relative Target Address for the  
4931 command target term. If the Resultant Address is not a UFiP, implementations shall return a Command  
4932 Status of COMMAND PROCESSING ERROR and a Processing Error of INVALID TARGET.

4933 The behavior of state-change commands for each UFcT is defined in DSP0216.

### 4934 6.12.3 Options

4935 Following are valid options for the `start` command in addition to those specified in **Error! Reference**  
4936 **source not found.**:

4937        **-f, -force**       Forces the implementation to start the object, ignoring any policy that might  
4938                               cause the implementation to normally not execute the command. The  
4939                               implementation shall execute this start if at all possible, without regard to  
4940                               consequences.

### 4941 6.12.4 Output

4942 This clause details requirements for CLP output for the `start` verb.

#### 4943 6.12.4.1 Text Format

4944 Implementations shall return Command Result data that includes the target address that was started (if  
4945 any) and the time and date when the start was initiated. Implementations may return the time/date  
4946 information in any applicable format that meets their needs. If the implementation did not start the target,  
4947 the implementation shall return Command Response data indicating that no target was started.

#### 4948 6.12.4.2 Structured Format

4949 This clause details requirements for structured output formats for the `start` verb.

##### 4950 6.12.4.2.1 General

4951 Implementations shall include any status data in the standard format at the top of the response. If the  
4952 target was successfully started, the `start` command shall then return the target started and the time the  
4953 start completed.

##### 4954 6.12.4.2.2 XML Output

4955 The implementation shall return the `start` element in the `response` element as defined in the  
4956 Command Response schema in DSP0224.

```
4957 <start>  
4958   <ufip>UFiP of target to start</ufip>  
4959   <timestamp>Time reset occurred if completed synchronously, returned in CIM  
4960     datetime format</timestamp>  
4961 </start>
```

##### 4962 6.12.4.2.3 Keyword

4963 Implementations shall use the following form when returning Command Results for the `start` command  
4964 in "keyword" format:

```
4965 command=start  
4966 ufip=User Friendly instance path of the Managed Element  
4967 timestamp= Time reset occurred if completed synchronously, returned in CIM datetime  
4968 format
```

4969     endoutput

## 4970     6.12.5 Examples

4971     This clause provides examples of the use of the `start` verb.

4972     EXAMPLE 1:     Sends `system1` to its default enabled state.

```
4973     -> start /system1
4974     /system1 started at 10:40am 1/1/01
```

4975     EXAMPLE 2:     Fails to start `cpu2` because this command is not supported on the target.

```
4976     -> start -o format=clpxml /system1/cpu2
4977     <?xml version="1.0" encoding="UTF-8"?>
4978     <response
4979     xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
4980     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4981     xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
4982     /dsp0224.xsd smclp_command_response.xsd">
4983         <command>
4984             <inputline>start -o format=clpxml /system1/cpu2</inputline>
4985         </command>
4986         <cmdstat>
4987             <status>3</status>
4988             <job>
4989                 <job_id>234</job_id>
4990                 <joberr>
4991                     <errtype>1</errtype>
4992                     <cimstat>7</cimstat>
4993                     <severity>2</severity>
4994                 </joberr>
4995             </job>
4996         </cmdstat>
4997         <start>
4998             <instance>
4999                 <ufit ufct="cpu" instance="2">cpu2</ufit>
5000                 <ufip>/system1/cpu2</ufip>
5001             </instance>
5002         </start>
5003     </response>
```

5004     EXAMPLE 3:     Fails to start `cpu2` because this command is not supported on the target.

```
5005     -> start -o format=keyword /system1/cpu1
5006     commandline=start -o format=keyword /system1/cpu2
5007     status=3
5008     job_id=234
5009     errtype=1
5010     cimstat=7
5011     severity=2
5012     command=start
5013     ufip=/system1/cpu2
5014     endoutput
```

## Server Management Command Line Protocol (SM CLP) Specification

5015 EXAMPLE 4: Starts `system1` and waits for the job to complete.

```
5016 -> start -o format=clpxml -w /system1
5017 <?xml version="1.0" encoding="UTF-8"?>
5018 <response
5019 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
5020 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5021 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
5022 /dsp0224.xsd smclp_command_response.xsd">
5023   <command>
5024     <inputline>start -o format=clpxml -w /system1</inputline>
5025   </command>
5026   <cmdstat>
5027     <status>3</status>
5028     <job>
5029       <job_id>234</job_id>
5030     </job>
5031   </cmdstat>
5032   <start>
5033     <instance>
5034       <ufit ufct="system" instance="1">system1</ufit>
5035       <ufip>/system1</ufip>
5036     </instance>
5037     <timestamp>20050130145904.000000-300</timestamp>
5038   </start>
5039 </response>
```

5040 EXAMPLE 5: Starts `system1` and waits for the job to complete.

```
5041 -> start -w -o format=keyword /system1
5042 commandline=start -w -o format=keyword /system1
5043 status=0
5044 command=start
5045 ufip=/system1
5046 timestamp=20050130145904.000000-300
5047 endoutput
```

### 5048 6.13 stop

5049 The general form of the `stop` command is:

```
5050 stop [<options>] [<target>]
```

#### 5051 6.13.1 General

5052 For the `stop` command, implementations shall support the syntax defined for the `stop-cmd` term in the  
5053 CLP grammar defined in Annex A.

5054 The `stop` command stops the target. If the target is already stopped, an error might or might not be  
5055 returned. The precise behavior is profile specific.

5056 This command can be used with and without Command Line options.



5057 **6.13.2 Valid Targets**

5058 This command is supported when it is specified in a target mapping (DSP0216) for a profile that the  
 5059 implementation supports. For all targets that do not support the use of the `stop` command,  
 5060 implementations shall not show the `stop` command in a command listing as being available.  
 5061 Implementations of the `stop` command will accept an Absolute or a Relative Target Address for the  
 5062 command target term. If the Resultant Address is not a UFiP, implementations shall return a Command  
 5063 Status of COMMAND PROCESSING ERROR and a Processing Error of INVALID TARGET.

5064 The behavior of state-change commands for each UFcT is defined in DSP0216.

5065 **6.13.3 Options**

5066 Following are valid options for the `stop` command in addition to those specified in **Error! Reference**  
 5067 **source not found.:**

5068       **-f, -force**     Forces the implementation to stop the target, ignoring any policy that might cause  
 5069                           the implementation to normally not execute the command. The implementation  
 5070                           shall execute this stop if at all possible, without regard to consequences.

5071 **6.13.4 Output**

5072 This clause details requirements for CLP output for the `stop` verb.

5073 **6.13.4.1 Text Format**

5074 Implementations shall return Command Result data that includes the target address that was stopped (if  
 5075 any) and the time and date when the stop was initiated. Implementations may return the time/date  
 5076 information in any applicable format that meets their needs. If the implementation did not stop the target,  
 5077 the implementation shall return Command Response data indicating that no targets were stopped.

5078 **6.13.4.2 Structured Format**

5079 This clause details requirements for structured output formats for the `stop` verb.

5080 **6.13.4.2.1 General**

5081 Implementations shall include any status data in the standard format at the top of the response. If the  
 5082 target was successfully stopped, the implementation shall then return the target that was stopped and the  
 5083 time the stop completed.

5084 **6.13.4.2.2 XML Output**

5085 The implementation shall return the `stop` element in the `response` element as defined in the Command  
 5086 Response schema in DSP0224.

```
5087 <stop>
5088   <ufip> Target address of Managed Element to stop </ufip>
5089   <timestamp>Time stop occurred if completed synchronously, returned in CIM
5090     datetime format</timestamp>
5091 </stop>
```

5092 **6.13.4.2.3 Keyword**

5093 Implementations shall use the following form when returning Command Results for the `stop` command in  
 5094 "keyword" format:

```
5095 command=stop
5096 ufip=User Friendly instance path of the Managed Element
```

## Server Management Command Line Protocol (SM CLP) Specification

```
5097 timestamp=Time reset occurred if completed synchronously, returned in CIM datetime
5098 format
5099 endoutput
```

### 6.13.5 Examples

5101 This clause provides examples of the use of the `stop` verb.

5102 EXAMPLE 1: Sends `system1` to its default disabled state.

```
5103 -> stop /system1
5104 /system1 stopped at 2:59:04 January 30, 2005.
```

5105 EXAMPLE 2: Stops `system1`.

```
5106 -> stop -o format=clpxml /system1
5107 <?xml version="1.0" encoding="UTF-8"?>
5108 <response
5109 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
5110 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5111 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0/
5112 dsp0224.xsd smclp_command_response.xsd">
5113   <command>
5114     <inputline>stop -o format=clpxml /system1</inputline>
5115   </command>
5116   <cmdstat>
5117     <status>0</status>
5118     <job>
5119       <job_id>9834</job_id>
5120     </job>
5121   </cmdstat>
5122   <stop>
5123     <instance>
5124       <ufit ufct="system" instance="1">system1</ufit>
5125       <ufip>/system1</ufip>
5126     </instance>
5127     <timestamp>20050130145904.000000-300</timestamp>
5128   </stop>
5129 </response>
```

5130 EXAMPLE 3: Stops `system1`.

```
5131 -> stop -o format=keyword /system1
5132 commandline=stop /system1
5133 status=0
5134 job_id=9834
5135 command=stop
5136 ufip=/system1
5137 timestamp=20050130145904.000000-300
5138 endoutput
```

5139 EXAMPLE 4: Specifies the `Examine` option and stops `system1`.

```
5140 -> stop -x -o format=clpxml /system1
5141 <?xml version="1.0" encoding="UTF-8"?>
5142 <response
5143 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
5144 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

## Server Management Command Line Protocol (SM CLP) Specification

```
5145 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
5146 /dsp0224.xsd smclp_command_response.xsd">
5147   <command>
5148     <inputline>stop -x -o format=clpxml /system1</inputline>
5149   </command>
5150   <cmdstat>
5151     <status>0</status>
5152   </cmdstat>
5153   <stop>
5154     <examine>
5155       <text>If run without the examine option, this will stop
5156         system1.</text>
5157     </examine>
5158   </stop>
5159 </response>
```

5160 EXAMPLE 5: Specifies the Examine option and stops system1.

```
5161 -> stop -x -o format=keyword /system1
5162 commandline=stop -x -o format=keyword /system1
5163 status=0
5164 job_id=923
5165 command=stop
5166 examine=If run without the examine option, this will stop system1.
5167 endoutput
```

5168 EXAMPLE 6: Attempt to stop system1 does not complete synchronously.

```
5169 -> stop /system1
5170 Stopping system1, job id is 9834.
```

5171 EXAMPLE 7: Attempt to stop system1 does not complete synchronously.

```
5172 -> stop -o format=clpxml /system1
5173 <?xml version="1.0" encoding="UTF-8"?>
5174 <response
5175 xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
5176 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5177 xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
5178 /dsp0224.xsd smclp_command_response.xsd">
5179   <command>
5180     <inputline>stop -o format=clpxml</inputline>
5181   </command>
5182   <cmdstat>
5183     <status>0</status>
5184     <job>
5185       <job_id>9834</job_id>
5186     </job>
5187   </cmdstat>
5188   <stop>
5189     <instance>
5190       <ufit ufct="system" instance="1">system1</ufit>
5191       <ufip>/system1</ufip>
5192     </instance>
5193   </stop>
5194 </response>
```

## Server Management Command Line Protocol (SM CLP) Specification

5195 EXAMPLE 8: Attempt to stop `system1` does not complete synchronously.

```
5196 -> stop -o format=keyword /system1
5197 commandline=stop -o format=keyword /system1
5198 status=1
5199 job_id=9834
5200 command=stop
5201 ufip=/system1
5202 endoutput
```

### 5203 6.14 version

5204 The general form of the `version` command is:

```
5205 version [<options>]
```

#### 5206 6.14.1 General

5207 For the `version` command, implementations shall support the syntax defined for the `version-cmd` term  
5208 in the CLP grammar defined in Annex A.

5209 The `version` command is used to display the version of the SM CLP that this implementation supports.  
5210 The implementation shall return the version of the SM CLP with which it is compatible.

#### 5211 6.14.2 Valid Targets

5212 The `version` command does not accept a command target term. Implementations shall not accept non-  
5213 option terms for the `version` command. If the implementation receives a command with non-option  
5214 terms specified, the implementation shall return a Command Status of COMMAND PROCESSING  
5215 ERROR and a Processing Error of COMMAND SYNTAX ERROR.

#### 5216 6.14.3 Options

5217 Valid options for the `version` command are specified in **Error! Reference source not found.**

#### 5218 6.14.4 Output

5219 This clause details requirements for CLP output for the `version` verb.

##### 5220 6.14.4.1 Text Format

5221 The implementation shall include the string "Version 1.0.0" clearly identified as the CLP version  
5222 supported by the implementation. The implementation shall include the version of the *Server*  
5223 *Management Managed Element (SM ME) Addressing Specification* with which it is conformant.

##### 5224 6.14.4.2 Structured Format

5225 This clause details requirements for structured output formats for the `version` verb.

###### 5226 6.14.4.2.1 General

5227 The returned data shall include any status data in the standard format at the top of the response.

###### 5228 6.14.4.2.2 XML Output

5229 The implementation shall return the `version` element in the `response` element as defined in the  
5230 Command Response schema in DSP0224. The implementation shall return the exact string "v1.0.0" as  
5231 the data contained in the `clpversion` element within the `version` element.

5232 **6.14.4.2.3 Keyword**

5233 Implementations shall use the following form when returning Command Results for the `version`  
5234 command in "keyword" format:

```
5235     command=version
5236     clpversion=v1.0.0
5237     addressingversion=v1.0.0
5238     endoutput
```

5239 **6.14.5 Examples**

5240 This clause provides examples of the use of the `version` verb.

5241 **EXAMPLE 1:** Runs the `version` command the way it is meant to be (that is, without targets).

```
5242     -> version
5243     SM CLP Version 1.0.0
5244     SM ME Addressing Version 1.0.0
```

5245 **EXAMPLE 2:** Attempts to include targets with the `version` command, but the `version` command does not  
5246 support targets.

```
5247     -> version /system1
5248     Invalid syntax; the version command does not support a target.
```

5249 **EXAMPLE 3:** Specifies invalid syntax for the `version` command.

```
5250     -> version -o format=clpxml /system1
5251     <?xml version="1.0" encoding="UTF-8"?>
5252     <response xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
5253     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5254     xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
5255     /dsp0224.xsd smclp_command_response.xsd">
5256         <command>
5257             <inputline>version -o format=clpxml /system1</inputline>
5258         </command>
5259         <cmdstat>
5260             <status>2</status>
5261             <error>252</error>
5262             <error_tag>COMMAND SYNTAX ERROR</error_tag>
5263         </cmdstat>
5264         <version></version>
5265     </response>
```

5266 **EXAMPLE 4:** Successfully runs the `version` command.

```
5267     -> version -o format=clpxml
5268     <?xml version="1.0" encoding="UTF-8"?>
5269     <response
5270     xmlns="http://schemas.dmtf.org/smash/commandresponse/1.0.0/dsp0224.xsd"
5271     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5272     xsi:schemaLocation="http://schemas.dmtf.org/smash/commandresponse/1.0.0
5273     /dsp0224.xsd smclp_command_response.xsd">
5274         <command>
5275             <inputline>version -o format=clpxml</inputline>
5276         </command>
5277         <cmdstat>
5278             <status>0</status>
5279         <job>
```

## Server Management Command Line Protocol (SM CLP) Specification

```
5280         <job_id>930</job_id>
5281         </job>
5282     </cmdstat>
5283     <version>
5284         <clpversion>v1.0.0</clpversion>
5285         <addressingversion>v1.0.0</addressingversion>
5286     </version>
5287 </response>
```

5288 EXAMPLE 5: Successfully runs the `version` command.

```
5289 -> version -o format=keyword
5290 command=version -o format=keyword
5291 clpversion=v1.0.0
5292 addressingversion=v1.0.0
5293 endoutput
```

## 5294 7 Standard Command Options

5295 This clause describes the requirements for supporting and implementing standard command options.

### 5296 7.1 General

5297 Every CLP option is defined to have the same name and behavior across the CLP command verb set.  
5298 For each CLP verb, implementations shall recognize the CLP standard options applicable to the verb and  
5299 observe the specified behavior of each option in a manner consistent with the option definition.

5300 EXAMPLE 1: Use of "-o" always indicates the `output` option whenever used and follows the behavior specified for  
5301 the `output` standard option.

5302 The clauses below describe each standard option and its behavior.

5303 Implementations shall recognize option names both by the full expression of the option name and by a  
5304 single letter short form of the option name if one is defined by this specification. Implementations shall not  
5305 recognize option names of any other length. That is, "partial" or "two letter" option names are not allowed.

5306 EXAMPLE 2: The option name for controlling output is recognized by its full option name form, "-output", and by its  
5307 single character short form, "-o", and not by any other form, such as "-out".

5308 For each CLP Command processed, implementations shall observe the default setting of standard  
5309 options unless the user overrides the setting with a session default or by using the standard option  
5310 explicitly in the Command Line entered. For each CLP Command processed, implementations shall apply  
5311 every standard option that is supported for the verb. Implementations shall observe the following order of  
5312 precedence for determining the value of a standard option: specification default, superseded by the  
5313 session default, and finally superseded by the explicit Command Line setting, if present.

5314 Implementations shall observe the following order of precedence for the following standard options: `help`,  
5315 `version`, and `examine`. If the `help` option appears on the Command Line, implementations shall  
5316 provide the behavior specified in **Error! Reference source not found..** If the `version` option appears,  
5317 but the `help` option does not, implementations shall provide the behavior specified in **Error! Reference**  
5318 **source not found..** If the `examine` option appears on the Command Line without either the `help` or  
5319 `version` option, the implementation shall provide the behavior described in **Error! Reference source**  
5320 **not found..** These precedence rules do not affect the processing of standard options or other options  
5321 specified in the Command Line.

5322 If an option is included more than once in a command, the implementation shall not execute the  
5323 command. The implementation shall return Command Response data indicating a **COMMAND SYNTAX**  
5324 **ERROR.**

## Server Management Command Line Protocol (SM CLP) Specification

5325 CLP options either always require an argument or never require an argument. If a CLP option that  
 5326 requires an argument is specified in a command without an argument value, the implementation shall  
 5327 return a Command Status of COMMAND PROCESSING FAILED and a Processing Error of MISSING  
 5328 ARGUMENT. If a CLP option that does not require an argument is specified in a command with an  
 5329 argument value, the implementation shall return a Command Status of COMMAND PROCESSING  
 5330 FAILED and a Processing Error of COMMAND SYNTAX ERROR. If a CLP option is specified with an  
 5331 argument that is not defined by the CLP as a legal argument and the argument is not identified as an  
 5332 OEM-defined argument according to the rules in 5.2.6.2, the implementation shall return a Command  
 5333 Status of COMMAND PROCESSING FAILED and a Processing Error of INVALID ARGUMENT.

5334 Implementations shall support only those options listed in Table 14. Implementations shall support all of  
 5335 the options listed in Table 14. Implementations shall support arguments on all options that specify an  
 5336 argument in Table 14. Implementations shall not support arguments for those options that do not specify  
 5337 an argument in Table 14.

5338 For each option listed in Table 14 that has a value of "yes" in the column labeled "Session Default  
 5339 Supported", the implementation shall allow the user to set a default value per session. For each option  
 5340 listed in Table 14 that does not have a value of "yes" in the column labeled "Session Default Supported",  
 5341 the implementation shall not allow the user to set a default value per session. Each option with a  
 5342 supported session default value is modeled with properties on the CIM class that represents the CLP  
 5343 session. For information on the CIM class and its properties, see the *CLP Service Profile* (DSP1005).

5344 **Table 14 – Standard Command Options**

Option Name	Short Form	Interpretation	Argument	Session Default Supported
-all	-a	Instructs the verb to perform all of its possible functions.	None	
-destination <URI>	<none>	Indicates the location of a destination for an image or other target data.	URI or SM instance address	
-display	-d	Selects the data the user wants to display.	Multiple arguments controlling the type of information returned about a target.	yes
-examine	-x	Instructs the Command Processor to examine the command for syntax and semantic errors but not execute the command.	None	
-force	-f	Instructs the verb to ignore warning conditions that would prevent execution by default.	None	
-help	-h	Displays documentation about the command verb.	None	
-keep <m[.s]>	-k	Establishes a holding time for command job ID and status.	Amount of time to hold command job ID, status	yes
-level <n>	-l	Instructs the Command Processor to execute the command for the current target plus targets contained through the specified level of depth.	Number of levels expressed as a natural number or "all"	

## Server Management Command Line Protocol (SM CLP) Specification

Option Name	Short Form	Interpretation	Argument	Session Default Supported
-output <args>	-o	Controls the content and form of the command output.	Many arguments providing control of format, language, level of detail, order, and so on of output data.	yes
-source <URI>	<none>	Indicates the location of a source image or target.	URI or SM instance address	
-version	-v	Displays the version of the command verb.	None	
-wait	-w	Instructs the Command Processor to hold the Command Response until all spawned jobs have completed.	None	yes

5345 Table 15 **Error! Reference source not found.** specifies the requirements for implementations to support  
 5346 each combination of verb and standard option. Except where noted, a cell with an X indicates that the  
 5347 implementation shall support use of the option named in the row with the verb named in the column.

5348 **Table 15 – Verb and Option Support**

CLP Option	CLP Verb												
	cd	create	delete	dump	exit	help	load	reset	set	show	start	stop	version
all										X			
destination				X									
display										X			
examine	X	X	X	X	X	X	X	X	X	X	X	X	X
force			X <sup>2</sup>	X <sup>1</sup>			X <sup>1</sup>	X <sup>2</sup>	X <sup>2</sup>	X <sup>1</sup>	X <sup>2</sup>	X <sup>2</sup>	
help	X	X	X	X	X	X	X	X	X	X	X	X	X
keep	X	X	X	X	X	X	X	X	X	X	X	X	X
level										X			
output	X	X	X	X	X	X	X	X	X	X	X	X	X
source							X						
version	X	X	X	X	X	X	X	X	X	X	X	X	X
wait	X	X	X	X		X	X	X	X	X	X	X	X

**Notes:** X<sup>1</sup> – The implementation *may* support use of the force option with the dump, load, and show verbs.  
 X<sup>2</sup> – The implementation *should* support use of the force option with the delete, reset, set, start, and stop verbs.

5349 If the cell in Table 15 **Error! Reference source not found.** is blank, the implementation shall not support  
 5350 the use of the option named in the row with the verb named in the column. If a Command Line is specified  
 5351 with a verb and option combination that shall not be supported according to **Error! Reference source**  
 5352 **not found.**, the implementation shall not execute the command and shall return a Command Status of  
 5353 COMMAND PROCESSING FAILED and a Processing Error of INVALID OPTION. If the implementation  
 5354 supports an option that may or should be supported according to Table 15 **Error! Reference source not**



5355 **found.** for one verb, the implementation may support the option with other verbs that may or should be  
5356 supported according to Table 15**Error! Reference source not found.** If a Command Line is specified  
5357 with a verb and option combination that may or should be supported according to Table 15**Error!**  
5358 **Reference source not found.** and is not supported by the implementation, the implementation shall not  
5359 execute the command and shall return a Command Status of COMMAND PROCESSING FAILED and a  
5360 Processing Error of OPTION NOT SUPPORTED.

## 5361 7.2 all

5362 This clause describes the requirements for the `all` option.

### 5363 7.2.1 Forms

```
5364 -all  
5365 -a
```

### 5366 7.2.2 Example Use

```
5367 show -a log1
```

### 5368 7.2.3 Behavior

5369 The `all` option instructs the Command Processor to select all values where appropriate for the  
5370 command. This option is supported only with the `show` command. For a detailed description of its use,  
5371 see 6.11.3.

5372 When the `all` option is specified, the implementation will select all values where appropriate for the  
5373 command.

## 5374 7.3 destination

5375 This clause describes the requirements for the `destination` option.

### 5376 7.3.1 Forms

```
5377 -destination
```

### 5378 7.3.2 Example Use

```
5379 dump -destination  
5380 ftp://administrator:passw0rd@myserver.com/private/administrator/memory.dmp  
5381 /system1/memory1
```

### 5382 7.3.3 Behavior

5383 The `destination` option is used to specify a destination for the transfer of a binary image. The desired  
5384 destination is specified as the argument to the `destination` option. The destination can be expressed  
5385 as a UFIP if it is a Managed Element in the address space of the MAP. The destination can also be  
5386 expressed as a URI. The desired protocol to use for transferring the image can be specified as part of the  
5387 URI. This specification does not mandate support for a specific protocol. Thus the protocols supported  
5388 are implementation specific. Implementations shall interpret the value specified as a URI defined  
5389 according to *Uniform Resource Identifiers (URI): Generic Syntax* (RFC2396). The URI specified may  
5390 contain a scheme that indicates the explicit service and location to be used to capture the dumped data. If  
5391 the URI specified is relative, then the implementation shall interpret the URI according to the rules  
5392 specified in 5.1.3.3.

5393 **7.4 display**

5394 This clause describes the requirements for the `display` option.

5395 **7.4.1 Forms**

5396 `-d[isplay] <type>*[,<type>]`

5397 **7.4.2 Example Use**

5398 EXAMPLE 1: Shows the commands that are supported for use with `log1` as a target.

5399 `show -display verbs log1`

5400 EXAMPLE 2: Shows all of the systems in the address space. No filtering of the results for individual instances  
5401 is requested.

5402 `show -level all -display targets=system /`

5403 EXAMPLE 3: Shows all of the CPUs and SystemDevice association instances in `system1`. No filtering of the  
5404 results for individual instances is requested.

5405 `show -level all -display targets=cpu,associations=systemdevice`  
5406 /system1`

5407 EXAMPLE 4: Shows the OperationalStatus property for every power supply and processor in `system1`.

5408 `show -level all -display ` targets=(cpu,pwrsupply),properties=  
5409 operationalstatus ` /system1`

5410 EXAMPLE 5: Shows the ElementName property for every instance.

5411 `show -level all -display properties=(elementname) /`

5412 **7.4.3 Behavior**

5413 The `display` option filters the information returned in Command Results. This option provides two levels  
5414 of filtering. It enables filtering of results for entire instances at the granularity of CIM class. The `display`  
5415 option also enables filtering results for an individual instance, allowing a user to restrict the results to the  
5416 properties (or a subset) of the instance or the verbs (or a subset) supported for an instance. For any two  
5417 commands that are identical except for arguments to the `display` option, the implementation shall  
5418 interpret and execute the command such that the set of Targets affected, and the resultant state of those  
5419 Targets, is identical. The `display` option requires one or more arguments specifying the category of  
5420 information to include in the Command Results. For each command, the implementation shall not include  
5421 data in the Command Results returned to the Client unless it is specified by the arguments for the  
5422 `display` option that are in effect for the command.

5423 The valid types and formats for the arguments of the `display` option are as follows:

5424 **all** Perform no filtering of the results.

5425 **verbs** Display the commands that are valid for this target.

5426 **properties**["(" <name>,<name>,...,<name>)" ]

5427 Can be used to filter the Command Results such that information about an instance is  
5428 returned only if the instance has a property with the specified value or a property with the  
5429 specified name. Can also be used to filter the Command Results so that only those  
5430 properties specified by name are returned. The properties are returned in the order  
5431 indicated. If no property names are specified, all properties included in the Command  
5432 Results will be returned.

5433 **targets**["(" (UFCt)[,<UFCt>,<UFCt>,...,<UFCt>)" ]

## Server Management Command Line Protocol (SM CLP) Specification

5434 Filter the Command Results to include information about Managed Element instances only.

5435 `associations=["(classname)[,<classname>,<classname>,...,<classname>"]`

5436 Filter the Command Results to include information about associations only.

5437 The `display` option acts as a filter on output of a command. The results of a command are marshaled  
5438 into operation results data. Arguments to the `display` option are used to select which information is  
5439 included in the Command Results data. Arguments to the `display` option are applied in the following  
5440 order:

- 5441 • `targets` (instance selection by class)
- 5442 • `associations` (instance selection by class)
- 5443 • `properties` (restrict to just properties returned for an instance)
- 5444 • `properties` (`propname`, restrict properties that are returned for an instance)
- 5445 • `verbs`

5446 The specification default argument for the `display` option is `targets,properties,verbs`.

5447 When the `display` option is specified with one or more arguments, the implementation shall apply each  
5448 of the arguments to the operation results according to the rules for each argument. For each argument  
5449 present in the Command Line, implementations shall apply the arguments in the following order  
5450 irrespective of the order in which they appear in the Command Line: `targets`, `associations`, `properties`,  
5451 `verbs`. When the `display` option is specified with an argument of `all`, the implementation shall interpret  
5452 the `display` option as if it was specified with an argument of  
5453 `"targets,associations,verbs,properties"`. Prior to processing the argument values, the  
5454 implementation shall remove the leading and trailing parentheses, if they are present. The filtering  
5455 provided for by the `display` option is applied across the entire set of Managed Elements and  
5456 Associations for which results were generated by the command. For example, when the `show` command  
5457 is used with the `level` option or with the Target Class Addressing, a tree of Managed Elements and  
5458 Associations can be returned. The filtering would then be applied across each level of the tree, removing  
5459 any Managed Elements that did not match the target filter, removing any Associations that did not match  
5460 the association filter, filtering properties returned on any remaining Managed Elements and Associations  
5461 according to the properties filter, and filtering the list of verbs supported for any remaining Managed  
5462 Elements. Note that for an individual Association instance, the `verbs` argument has no effect because  
5463 Command Results for an instance of an Association do not include information about the verbs supported  
5464 for an instance. Implementations will accept duplicate values within the comma-delimited list of values for  
5465 each argument to the `display` option.

5466 When the `display` option is specified with an argument of `targets` and no value is supplied to the  
5467 argument, the implementation shall filter the operation results and add to the Command Results  
5468 information about Managed Element instances. If the `display` option is specified with an argument of  
5469 `targets` and a value is supplied to the `targets` argument, the implementation shall interpret the  
5470 supplied value as a comma-delimited, ordered list of UFcTs. The implementation shall apply the ordered  
5471 list of UFcTs as a filter on the instances for which results are returned. The implementation shall add  
5472 operation results for an instance if and only if the instance is of one of the types specified by a UFcT in  
5473 the list. The implementation shall add the results for instances in the order their corresponding types were  
5474 specified in the UFcT list. When selecting instances for which results will be included in the Command  
5475 Results, the implementation shall apply the "Rules for Selecting Instances by UFcT" defined in DSP0215  
5476 for each UFcT supplied as a value to the `targets` argument and each instance whose results are  
5477 included in the operation results, where the Selection UFcT is the UFcT supplied as an argument value  
5478 and the target instance is the instance whose results are included in the operation results.

5479 When the `display` option is specified with an argument of `associations` and no value is supplied to  
5480 the argument, the implementation shall filter the operation results and add to the Command Results

## Server Management Command Line Protocol (SM CLP) Specification

5481 information about each Association. If the `display` option is specified with an argument of  
5482 `associations` and a value is supplied to the `associations` argument, the implementation shall  
5483 interpret the supplied value as a comma-delimited, ordered list of Association Classes. The  
5484 implementation shall apply the ordered list of Association Classes as a filter on the instances for which  
5485 results are returned. The implementation shall add operation results for an instance if and only if the  
5486 instance is of one of the types specified by an Association Class in the list. The implementation shall add  
5487 the results for instances in the order their corresponding types were specified in the list of class names.

5488 The `properties` argument to the `display` option can be used to filter the results returned for a target  
5489 to be the properties of the target or a specific subset of the properties. Usage is supported with or without  
5490 additional values. When the `display` option is specified with an argument of `properties`,  
5491 implementations shall apply the filtering of the `properties` argument to the intermediate results that are  
5492 generated by applying the filtering of the `association` and `targets` arguments to the operation  
5493 results. If the `display` option is specified with an argument of `properties` and a value is supplied to  
5494 the `properties` argument, the implementation shall interpret the value as a comma-delimited, ordered  
5495 list of tokens.

5496 The implementation shall include information about a property of an instance, if the property name  
5497 appears in the ordered list of property names that were specified as the value of the `properties`  
5498 argument.

5499 When the `display` option is specified with an argument of `properties` and no value is supplied to the  
5500 argument, the implementation shall filter the intermediate results and add to the Command Results for an  
5501 instance information about the properties of the instance.

5502 When the `display` option is specified with an argument of `verbs`, the implementation shall filter the  
5503 intermediate results and add to the Command Results for an instance information about the verbs that  
5504 are supported for the instance.

5505 It is possible that applying the filters on information returned for an instance will result in there being no  
5506 information to return for an instance. After applying the filters, if there is no information for an instance,  
5507 the implementation shall remove the instance from the Command Results that are returned.

### 5508 **7.5 examine**

5509 This clause describes the requirements for the `examine` option.

#### 5510 **7.5.1 Forms**

```
5511 -examine  
5512 -x
```

#### 5513 **7.5.2 Example Use**

```
5514 stop -force -x /system3
```

#### 5515 **7.5.3 Behavior**

5516 The `examine` option causes the Command Processor to examine command syntax only and provide  
5517 feedback to the user on the command's validity and correctness. If the `examine` option is included in the  
5518 Command Line, the implementation will perform the standard validation of the Command Line but will not  
5519 execute the command. Thus, any errors that would result in a Command Status of `COMMAND`  
5520 `PROCESSING FAILED` and the accompanying Processing Error if the `examine` option were not specified  
5521 will still result in a Command Status of `COMMAND PROCESSING FAILED` and the accompanying  
5522 Processing Error. Any errors that would result in a `COMMAND EXECUTION FAILED` will be reported as  
5523 free-form text in the Command Results and not as a Command Status.

5524 When the `examine` option is included in the Command Line, the Command Processor shall not commit  
5525 or execute the command. The Command Processor shall verify that the options, Resultant Target, and  
5526 target properties are valid. The Command Results may describe what action would be taken if the  
5527 command were to be entered without the `examine` option. Although the Command Processor will not  
5528 execute the command, applying the `examine` option to the Command Line is itself an activity of the MAP  
5529 that will result in a Job being created.

### 5530 **7.6 force**

5531 This clause describes the requirements for the `force` option.

#### 5532 **7.6.1 Forms**

5533 `-force`

5534 `-f`

#### 5535 **7.6.2 Example Use**

5536 `stop -f /system3/blade2`

#### 5537 **7.6.3 Behavior**

5538 The `force` option causes the Command Processor to ensure that the command executes, regardless of  
5539 potential preparation steps that could have been taken, initial ME states recommended, or exceptions that  
5540 occur during execution. The `force` option does not override authorization of a user to execute a  
5541 command for a given target.

5542 When the `force` option is specified, the implementation shall execute the command.

### 5543 **7.7 help**

5544 This clause describes the requirements for the `help` option.

#### 5545 **7.7.1 Forms**

5546 `-help`

5547 `-h`

#### 5548 **7.7.2 Example Use**

5549 `stop -help`

#### 5550 **7.7.3 Behavior**

5551 The `help` option causes the Command Processor to return text describing the proper use of the verb  
5552 entered as the first term on the Command Line. The help text includes a description of the verb and its  
5553 behavior and descriptions of all options supported by the verb. When the `help` option is specified, the  
5554 command does not execute or cause any change to the target. It does not alter the state or properties of  
5555 a target that appear on the same Command Line. Help output text is governed by the language option  
5556 currently in effect.

5557 When the `help` option is specified, the implementation shall not take the action specified by the verb. The  
5558 implementation shall return text describing the proper use of the verb entered as the first term on the  
5559 Command Line.

## Server Management Command Line Protocol (SM CLP) Specification

### 5560 7.8 keep

5561 This clause describes the requirements for the `keep` option.

#### 5562 7.8.1 Forms

```
5563 -keep <m[.s]>
```

```
5564 -k <m[.s]>
```

5565 *m.s* is an amount of time specified as 'minutes.seconds'.

#### 5566 7.8.2 Example Use

```
5567 reset -k 10 system1
```

5568 Resets `system1` and retains the Command Status for 10 minutes.

#### 5569 7.8.3 Behavior

5570 The `keep` option is used to request the Command Processor to retain status information for the job  
5571 spawned in response to the command with which the option was included. In order to use the `keep`  
5572 option to request that the status for a command be preserved, it must be included as part of the initial  
5573 command request.

5574 Implementations shall keep the `CIM_ConcreteJob` instance corresponding to the job in the job queue  
5575 after the job is completed for the time specified by the `keep` option. The implementation shall set the  
5576 `TimeBeforeRemoval` property of the `CIM_ConcreteJob` instance to the value specified by the `keep`  
5577 option. Implementations may maintain Command Results after the command execution completes even if  
5578 the `keep` option is used. The `keep` option only controls how long the implementation is required to  
5579 maintain the Command Status.

5580 If the CLP session ends before the command has completed and the Command Status has been  
5581 returned, the Command Status will be retained either for the amount of time that was specified with the  
5582 `keep` option or the session default, as appropriate.

### 5583 7.9 level

5584 This clause describes the requirements for the `level` option.

#### 5585 7.9.1 Forms

```
5586 -level <n>
```

```
5587 -l <n>
```

5588 *n* is the number of levels to include in the command scope.

#### 5589 7.9.2 Example Use

5590 EXAMPLE 1: Shows information about the default target and one level of contained Managed Elements.

```
5591 show -l 2
```

5592 EXAMPLE 2: Shows information about Managed Elements up to two levels deep.

```
5593 show -l 3
```

5594 EXAMPLE 3: Stops all contained Managed Elements, and then stops the command target.

```
5595 stop -l all
```

5596 EXAMPLE 4: Shows information about `system3` and all contained Managed Elements.

5597 `show -l all system3`

5598 **7.9.3 Behavior**

5599 The `level` option instructs the command verb to include *n* number of levels in the scope of its execution.  
 5600 A level typically refers to the depth of containment that is to be processed by the verb, following the target  
 5601 addressing syntax described in 5.2.1.3.5.

5602 Any levels specified that extend beyond the containment depth of the command target shall be ignored.  
 5603 For example, if a command target contains only one level of contained targets and the user attempts to  
 5604 show "n=5" levels, the command is still executed successfully and the one level of containment is  
 5605 returned in the output.

5606 The value of *n* is interpreted as follows:

5607 **n=1** Verb is interpreted for the command target only (default).

5608 **n=2** Verb acts on the command target and any directly contained Managed Elements.

5609 **n=3** Verb acts on the command target, directly contained Managed Elements, and any  
 5610 Managed Elements contained by those Managed Elements (that is, the current target and  
 5611 "two down").

5612 **n=all** Verb acts on the command target and all target Managed Elements recursively contained  
 5613 in the command target.

5614 When the `level` option is included with a command, the implementation shall apply the command to  
 5615 each level of containment up to the minimum of the level of containment specified by the argument to the  
 5616 level option and the actual containment depth of the command target.

5617 **7.10 output**

5618 This clause describes the requirements for the `output` option.

5619 **7.10.1 Forms**

5620 `-output <arguments>`

5621 `-o <arguments>`

5622 Table 16 lists the arguments for the output option.

5623 **Table 16 – Output Option Arguments**

Argument	Value Domain (default in bold)	Description
<code>format=&lt;value&gt;</code>	<b>"text"</b> , "keyword", "clpxml"	<code>format</code> controls the structure of the output text.
<code>error</code> , <b><code>terse</code></b> , <code>verbose</code>		This argument selects the level of detail included in the output.
<code>language=&lt;value&gt;</code>	3-character string identifier of language as specified in ISO 639.2; <b>"eng"</b> is default.	<code>language</code> selects the translation of text.
<b><code>begin</code></b> , <code>end</code>		When multiple items are returned in the output, <code>begin</code> and <code>end</code> control where to start in the list.

## Server Management Command Line Protocol (SM CLP) Specification

Argument	Value Domain (default in bold)	Description
<code>order=&lt;value&gt;</code>	<b>"default"</b> , "reverse"	When multiple items are returned in the output, <code>order</code> controls the order of those items.
<code>count=&lt;value&gt;</code>	<integer string or <b>"all"</b> >	When multiple items are returned in the output, <code>count</code> controls the number of items returned (for example, log items); the default is 'all' items.  The maximum value for <code>count</code> is determined by the class of the target.
<code>number=&lt;x-y&gt;</code>	[integer string]- [integer string]	<code>number</code> requests that a range of results be returned.

### 5624 7.10.2 Example Use

```
5625 show -output format=clpxml,verbose,order=reverse,number=5 ./log1/record*
5626 show -o verbose /system2/log1
5627 start -o terse system1
```

### 5628 7.10.3 Behavior

5629 The `output` option controls the format of output returned by the Command Processor to the Client. This  
5630 option has no effect on the execution of the command or the Targets affected by the command.

#### 5631 7.10.3.1 General Requirements

5632 For any two commands that are identical except for arguments to the `output` option, the implementation  
5633 shall interpret and execute the command such that the Targets affected, and the resultant state of those  
5634 Targets, is identical.

5635 EXAMPLE: The following command would cause all of the instances of the UFcT in the container to be  
5636 deleted, not just four of them.

```
5637 delete -output count=4 UFcT
```

5638 Prior to processing the value for an argument, the implementation shall remove the leading and trailing  
5639 parentheses if they are present. For each combination of arguments to the `output` option,  
5640 implementations shall return identical Command Results irrespective of the relative order in which the  
5641 arguments appear on the Command Line.

5642 Implementations shall return Command Results consistent with applying arguments to the `output` option  
5643 in the following order: `number`, `begin`, `end`, `count`, `order`, `format`, `terse`, `error`, `verbose`, and  
5644 `language`.

#### 5645 7.10.3.2 Output Format Selection

5646 By default, all output is returned in free-form English text, which is not suitable for parsing. The user may  
5647 request that output be formatted in a structured form by using the `format` argument and an associated  
5648 value representing the desired structure. For example, if the user wants the output to be returned in an  
5649 XML document format, the option form `-output format=clpxml` is used.

5650 If the `output` option with the `format` argument is included in a command, the implementation shall  
5651 attempt to interpret the value of the `format` argument as one of the CLP output format identifiers. If the  
5652 value of the `format` argument is a CLP output format identifier and the implementation supports the  
5653 indicated output format, the implementation shall return Command Response data compliant with the  
5654 format specified. If the value of the `format` argument is a CLP output format identifier and the



5655 implementation does not support the indicated output format, the implementation shall not execute the  
5656 command and the implementation shall return Command Response data compliant with the session  
5657 default output format indicating OUTPUT FORMAT NOT SUPPORTED. If the argument value is not a  
5658 CLP output format identifier, the implementation shall not execute the command and the implementation  
5659 shall return Command Response data compliant with session default output format indicating a  
5660 Processing Error of INVALID ARGUMENT.

5661 Implementations shall recognize the string `text` as identifying the CLP text output format and shall not  
5662 recognize any other string as identifying the text output format. Implementations shall recognize the string  
5663 `keyword` as identifying the CLP keyword/value output format and shall not recognize any other string as  
5664 identifying the keyword/value output format. Implementations shall recognize the string `clpxml` as  
5665 identifying the CLP XML output format and shall not recognize any other string as identifying the XML  
5666 output format.

### 5667 7.10.3.3 Output Language Selection

5668 If the `output` option with the language argument is included in a command, the implementation shall  
5669 attempt to interpret the value of the language argument as a language code defined in accordance with  
5670 ISO 639-2. If the implementation recognizes the value of the language argument as identifying an output  
5671 language it supports, the implementation shall return Command Response data in the language specified.  
5672 If the value of the language argument is not recognized by the implementation as identifying an output  
5673 language it supports, the implementation shall not execute the command, the implementation shall return  
5674 Command Response data in the session default language indicating a Processing Error of INVALID  
5675 ARGUMENT.

### 5676 7.10.3.4 Output Range and Order Selection

5677 Some CLP verbs can return Command Results that include results for more than one Managed Element.  
5678 The results of a command are marshaled into operation results data. The operation results data contains  
5679 the results for each Managed Element as an ordered list. For each Managed Element, DSP0216 defines  
5680 the algorithm for initially ordering the operation results into the natural (default) order. Note that  
5681 randomized is considered a valid algorithm for ordering the results. Arguments to the `output` option that  
5682 select a range of elements for which results will be returned or control the order in which results are  
5683 returned will operate against this list of operation results. The `begin` argument instruments the  
5684 implementation to select the range of elements for which results will be included from the beginning of the  
5685 list. The `end` argument instruments the implementation to select the range of elements for which results  
5686 will be included from the end of the list. The `count` argument allows a user to restrict the number of  
5687 Managed Elements in the range. The `order` argument allows a user to modify the order in which results  
5688 are returned for a command. The `number` argument allows a user to select a range of Managed  
5689 Elements.

5690 The `begin`, `end`, `number`, `order`, and `count` arguments to the `output` option affect the order and  
5691 range of results displayed. When operation results are returned for Managed Elements of more than one  
5692 type, the implementation shall apply the range selection once for each Managed Element type to the  
5693 range of instances of that type. Implementations shall not apply the range selection arguments within the  
5694 result for an individual Managed Element. Implementations shall apply the output order and range  
5695 selection arguments to the elements at the deepest level of the containment hierarchy for which operation  
5696 results have been generated and shall not apply the range or output order selection arguments to  
5697 instances at any other level of the containment hierarchy for which operation results were generated. The  
5698 `begin`, `end`, and `number` arguments are mutually exclusive. If the implementation receives a command  
5699 with the `output` option specified with more than one of `begin`, `end`, and `number` specified, the  
5700 implementation shall not execute the command and shall return a Command Status of COMMAND  
5701 PROCESSING FAILED with a Processing Error of COMMAND SYNTAX ERROR. The `number` and  
5702 `count` arguments are mutually exclusive. If the implementation receives a command with the `output`  
5703 option specified with more than one of `number` and `count` specified, the implementation shall not

## Server Management Command Line Protocol (SM CLP) Specification

5704 execute the command and shall return a Command Status of COMMAND PROCESSING FAILED with a  
5705 Processing Error of COMMAND SYNTAX ERROR.

5706 When the `output` option is specified with an argument of `begin`, the implementation shall use the first  
5707 element as the beginning element in the range of elements for which results will be returned. When the  
5708 `output` option is specified with an argument of `end`, the implementation shall use the last element as the  
5709 ending element in the range of elements for which results will be returned. Implementations shall then  
5710 apply the `count` argument to determine the number of elements for which operation results are included  
5711 in the Command Results. When the `output` option is specified with an argument of `count`, the  
5712 implementation shall not return results for more Managed Elements than the value specified for `count`.  
5713 Note that the value for the `count` argument is interpreted as an upper limit only. If the operation results  
5714 contain results for fewer Managed Elements than the value specified for the `count` argument, the `count`  
5715 argument will not have any effect.

5716 When the `output` option is specified with an argument of `order` and the value selected for `order` is  
5717 `default`, the implementation shall return results in the natural order for the class. When the `output`  
5718 option is specified with an argument of `order` and the value selected for `order` is `reverse`, the  
5719 implementation shall return results in reverse of the natural order for the class. If the `end` argument is not  
5720 specified, the implementation shall behave as if the `begin` argument was specified. If the  
5721 `order=reverse` argument is not specified, the implementation shall behave as if `order=default` was  
5722 specified. Thus the default behavior is to return elements according to their natural order.

5723 The `number` argument allows a user to select a range of Managed Elements. The value of the `number`  
5724 argument is in the following format:

5725 `<starting identifier>-<ending identifier>`

5726 Range selection is inclusive of the start and ending identifier. Any non-negative integer is an acceptable  
5727 value for the `starting identifier` and `ending identifier`. If the `number` argument is specified  
5728 with a value for the `starting identifier` or `ending identifier` that is not a non-negative  
5729 integer, the implementation shall return a Command Status of COMMAND PROCESSING FAILED and a  
5730 Processing Error of INVALID ARGUMENT. When the `output` option is specified with an argument of  
5731 `number`, the implementation shall select the range of instances identified by `starting identifier`  
5732 through the `ending identifier`, inclusive from the list of instances for which results are contained in  
5733 the operation results. If the value specified for the `starting identifier` is greater than the value  
5734 specified for the `ending identifier`, the implementation shall return a Command Status of  
5735 COMMAND PROCESSING FAILED and a Processing Error of INVALID ARGUMENT. If the value for the  
5736 `ending identifier` is greater than the number of elements in the list of operation results, the  
5737 implementation shall select the last element in the list of operation results as the end element in the range  
5738 for which results will be returned. If the value for the `starting identifier` is greater than the number  
5739 of elements in the list of operation results, the implementation shall not return results for any Managed  
5740 Elements.

5741 The following examples are provided for informational purposes and do not constitute additional  
5742 constraints on the implementation. For these examples, assume there are nine instances of a class  
5743 where the natural ordering of their results is `UFcT1`, `UFcT3`, `UFcT4`, `UFcT5`, `UFcT6`, `UFcT7`, `UFcT8`,  
5744 `UFcT9`, `UFcT10`. Note that the results do not contain an instance of `UFcT2` in this example.

5745 EXAMPLE 1:

5746 `- o begin`

5747 `begin` is the only argument to the `output` option. The implementation will select instance  
5748 `UFcT1` as the beginning element for the range of results. The `count` argument is not  
5749 specified, so the default value of "all" is used. The implementation will select all elements,  
5750 starting at `UFcT1`. The `order` argument is not specified, so the default ordering will be  
5751 used. Thus, results will be returned in the default ordering of `UFcT1`, `UFcT3`, `UFcT4`,  
5752 `UFcT5`, `UFcT6`, `UFcT7`, `UFcT8`, `UFcT9`, `UFcT10`.

5753 EXAMPLE 2:

5754 `-o begin,count=3`

5755 The implementation will select instance `UFcT1` as the beginning element for the range of  
 5756 results. The `count` is specified with a value of 3, so the implementation will select the first  
 5757 three instances. The `order` argument is not specified, so the default ordering will be used.  
 5758 Thus, results will be returned in the default ordering of `UFcT1`, `UFcT3`, `UFcT4`.

5759 EXAMPLE 3:

5760 `-o end`

5761 `end` is the only argument to the `output` option. The implementation will select instance  
 5762 `UFcT10` as the end element for the range of results. The `count` argument is not specified,  
 5763 so the default value of "all" is used. The implementation will select all elements, ending at  
 5764 `UFcT10`. The `order` argument is not specified, so the default ordering will be used. Thus,  
 5765 results will be returned in the default ordering of `UFcT1`, `UFcT3`, `UFcT4`, `UFcT5`, `UFcT6`,  
 5766 `UFcT7`, `UFcT8`, `UFcT9`, `UFcT10`.

5767 EXAMPLE 4:

5768 `-o end,count=3`

5769 `end` is specified, so the implementation will select instance `UFcT10` as the end element for  
 5770 the range of results. The `count` argument is specified with a value of 3, so the  
 5771 implementation will select the last three elements ending at `UFcT10`. The `order` argument  
 5772 is not specified, so the default ordering will be used. Thus, results will be returned in the  
 5773 default ordering of `UFcT8`, `UFcT9`, `UFcT10`.

5774 EXAMPLE 5:

5775 `-o end,order=reverse`

5776 As in EXAMPLE 3, the implementation will return results for all instances. However, they  
 5777 will be returned in the reverse order of `UFcT10...UFcT3`, `UFcT1`.

5778 EXAMPLE 6:

5779 `-o number=(1-5)`

5780 The implementation will select the first item in the list of operation results as the first  
 5781 element to include in the range of results. The implementation will select the fifth item in  
 5782 the list of operation results as the last element to include in the range of results. The  
 5783 implementation will include all of the elements in between. The results will be returned as  
 5784 `UFcT1`, `UFcT3`, `UFcT4`, `UFcT5`, `UFcT6`.

### 5785 7.10.3.5 Output Verbosity

5786 The `terse`, `verbose`, and `error` arguments are used to specify the level of detail an implementation  
 5787 returns when the output format is "text". If the implementation receives a command with the `output`  
 5788 option specified with more than one of the `terse`, `error`, and `verbose` arguments specified, the  
 5789 implementation shall not execute the command and shall return a Command Status of COMMAND  
 5790 PROCESSING FAILED with a Processing Error of COMMAND SYNTAX ERROR.

5791 The `terse`, `verbose`, and `error` arguments have no effect when the output format is a structured  
 5792 output format. When the output format in effect for a command is a structured output and the `terse`,  
 5793 `verbose`, and `error` arguments are in effect for a command, the `terse`, `verbose`, and `error`  
 5794 arguments are ignored.

## Server Management Command Line Protocol (SM CLP) Specification

5795 When the output format for a command is "text" and the `output` option is specified with an argument of  
5796 `terse`, the implementation shall return Command Status that includes only the Status Tag data element,  
5797 unless a processing error or execution error occurs, in which case the implementation shall return the  
5798 Processing Error Tag or Job Error data element as appropriate to the error that occurred.

5799 When the output format for a command is "text" and the `output` option is specified with an argument of  
5800 `error`, the implementation shall return Command Status if an error occurs in processing the command  
5801 and shall not return Command Status if an error does not occur in processing the command.

5802 When the output format for a command is "text" and the `output` option is specified with an argument of  
5803 `verbose`, the implementation shall return Command Status that includes the Status Tag. If a processing  
5804 error or execution error occurs, the implementation shall return the Processing Error Tag or Job Error  
5805 data element as appropriate to the error that occurred. The implementation may return Message data  
5806 elements.

5807 Table 17 lists the supported arguments for the output option that control the level of detail returned by a  
5808 command when the output format is text mode.

5809 **Table 17 – Output Options for Controlling Command Status Output**

Option	Description/Uses
<code>-o terse</code>	A short description of the status is returned.
<code>-o verbose</code>	Command Status and Command Results are returned.
<code>-o error</code>	Command Status is returned only if an error occurs.

### 5810 **7.11 source**

5811 This clause describes the requirements for the `source` option.

#### 5812 **7.11.1 Forms**

5813 `-source`

#### 5814 **7.11.2 Example Use**

```
5815 load -source `  
5816 ftp://administrator:passw0rd@myserver.com/private/administrator/`firmware.img  
5817 /system1/fw1
```

#### 5818 **7.11.3 Behavior**

5819 The `source` option is used to specify the source of data for a command. The source can be expressed  
5820 as a UFIIP if it is a Managed Element in the address space of the MAP. The source can also be expressed  
5821 as a URI. The desired protocol to use for transferring the image is specified as part of the URI. This  
5822 specification does not mandate support for a specific protocol. Thus the protocols supported are  
5823 implementation specific. Implementations shall interpret the value specified as a URI defined according to  
5824 RFC2396. The URI specified may contain a scheme that indicates the explicit service and location to be  
5825 used to transfer the source data. If the URI specified is relative, then the implementation shall interpret  
5826 the URI according to the rules specified in 5.1.3.3.

### 5827 **7.12 version**

5828 This clause describes the requirements for the `version` option.

5829 **7.12.1 Forms**

5830 `-version`  
 5831 `-v`

5832 **7.12.2 Example Use**

5833 EXAMPLE 1: Displays the version of the `start` command verb.

5834 `start -version`

5835 **7.12.3 Behavior**

5836 The `version` option causes the Command Processor to return the version of the command verb that  
 5837 appears as the first term on the Command Line. CLP command verbs are assigned the version of the  
 5838 CLP in which they were introduced or in which they were last modified. The CLP version tracks the overall  
 5839 version of the syntax and is revised whenever a new CLP verb is added to the standard or when an  
 5840 existing verb's syntax or semantics are revised. When the `version` option is used, the command verb  
 5841 itself is not executed.

5842 When the `version` option is included in a command, the implementation shall not execute the command.  
 5843 The implementation shall return Command Response data that identifies the version of the verb  
 5844 supported by the implementation.

5845 **7.13 wait**

5846 This clause describes the requirements for the `wait` option.

5847 **7.13.1 Forms**

5848 `-wait`  
 5849 `-w`

5850 **7.13.2 Example Use**

5851 `stop -w /system3`

5852 **7.13.3 Behavior**

5853 When the `wait` option is included in a command, the implementation shall not return control immediately  
 5854 to the Client; instead the implementation shall withhold all Command Response data and command input  
 5855 control until all jobs related to this command have completed. The `wait` option has no effect on the  
 5856 success or failure of the command or spawned jobs.

5857 **8 SM CLP Session**

5858 The following clauses describe the requirements for management of CLP sessions.

5859 **8.1 Authentication, Authorization, and Audit**

5860 The CLP Service relies on user accounts and user groups to control access to Managed Elements  
 5861 through the service. Membership in a user group conveys to the user account the management privileges  
 5862 associated with that user group. User accounts may be members of more than one user group. User  
 5863 privileges are additive—membership in a group conveys to the user all of the privileges of that group, not  
 5864 just the privileges that overlap with additional groups to which the user belongs.

## Server Management Command Line Protocol (SM CLP) Specification

5865 The CLP defines three user groups, each with an associated set of privileges. The three CLP-defined  
5866 user groups are Administrator, Operator, and Read Only. Implementations shall support the Read Only  
5867 and Administrator groups. Implementations should support the Operator group. Implementations may  
5868 support additional user groups.

5869 Authorization for the CLP-defined user groups is at SM CLP command granularity. If the user account for  
5870 the session is a member of a group that has permission to execute a command, the implementation shall  
5871 attempt to process the command. If the user account for the session is not a member of a group that has  
5872 permission to execute the command, the implementation shall not complete the requested command and  
5873 the implementation shall return a Command Status of COMMAND EXECUTION FAILED, a Job Error  
5874 Type of Security Error, and a CIM Status of CIM\_ERR\_ACCESS\_DENIED.

5875 Table 18 lists the CLP-defined user groups and the associated authorized CLP commands. For each user  
5876 group listed in the "Group" column, implementations shall authorize the CLP commands listed in the "SM  
5877 CLP Commands" column and shall not authorize CLP commands that do not appear in the "SM CLP  
5878 Commands" column.

5879 **Table 18 – Command Authorizations for CLP Groups**

Group	SM CLP Commands
Read Only	cd, exit, help ,show, version
Operator	cd, exit, help, show, version, reset, start, stop, set, load, dump
Administrator	cd, exit, help, show, version, reset, start, stop, set, load, dump, create, delete

5880 Prior to allowing a Client to issue any SM CLP commands, implementations shall have an authenticated  
5881 session established between the Client and the CLP Service. Every CLP session shall exist in the context  
5882 of a CLP User where the context is determined when the session is established and cannot be modified  
5883 for the session. Establishment of an authenticated session is implementation specific and is outside the  
5884 scope of this specification.

5885 Implementations shall support adding, removing, displaying, and modifying user accounts through the  
5886 CLP. Implementations shall require that a user belong to the Administrator group in order to create,  
5887 delete, or modify users and assign users to groups. This is an exception to the rules in Table 18. For  
5888 more information, see DSP0216. The representation of CLP users and groups is defined in DSP1005.  
5889 The definition of the initial user account for accessing the CLP is implementation specific and outside the  
5890 scope of this specification.

## 5891 **8.2 CLP Session**

5892 This clause describes the CLP session.

### 5893 **8.2.1 General**

5894 A CLP session is defined as a synchronous text message service exposed by a MAP CLP Service  
5895 through (or over) an underlying transport protocol. All commands and responses in a CLP session are  
5896 session data messages from the standpoint of the transport, and there are no required messages in a  
5897 CLP session except the termination command. A CLP session exists from the time the service is invoked  
5898 by one of the methods described in the corresponding transport mapping specification until termination is  
5899 requested by the Client by sending the CLP session termination command (*exit*).

5900 The CLP session also ends if the service terminates for any other reason. When a graceful stop of the  
5901 CLP session is initiated, the implementation shall not accept any additional commands from the user and  
5902 shall finish processing the pending command, if there is one. The implementation shall then end the  
5903 session as if it were processing an *exit* command from the user. If an immediate stop of the CLP

5904 session is initiated, the implementation shall immediately stop the CLP session and shall not return a  
5905 Command Response to the user. When terminating the CLP session, the implementation may also  
5906 terminate the transport session.

### 5907 **8.2.2 Session Environment**

5908 Interaction with the CLP Service is done within the context of a session. The CLP Service maintains a  
5909 session context for each active session. The SESSION term (see 5.1.7) references the session in which  
5910 the term is used. This provides a convenient mechanism for users to access their current session. The  
5911 CLP Service maintains useful information such as the Current Default Target, the default values for  
5912 options, the default target of the session, and the credentials provided when the session was initiated in  
5913 the session context. Session default values are supported only on a subset of options. For a complete list  
5914 of options and session attributes that implementations are required to support, see DSP0216.  
5915 Implementations shall allow users to modify settings for their own sessions irrespective of the groups of  
5916 which the users are a member. This is an exception to the rules in Table 18.

### 5917 **8.2.3 CLP Service**

5918 A CLP session is managed by the CLP Service. Within the scope of a single MAP/CLP Service, a user  
5919 may see other user sessions if the implementation supports multiple, simultaneous sessions, and if the  
5920 user has Administrator authorization.

5921 Implementations shall require that a user has Administrator authorization in order to show other user  
5922 sessions, stop other user sessions, or stop the CLP Service.

5923 If a graceful stop of the CLP Service is requested, the implementation shall gracefully stop each active  
5924 CLP session as described in 8.2.1 prior to ending the service. If an immediate stop of the CLP Service is  
5925 requested, the implementation shall immediately stop each active CLP session as described in 8.2.1 prior  
5926 to ending the service.

### 5927 **8.2.4 Initial Prompt**

5928 When a CLP session is established, implementations shall return the following string followed by the first  
5929 command prompt:

```
5930 === SM CLP v<major>.<minor>.<patchlevel> SM ME Addressing  
5931 v<major>.<minor>.<patchlevel> <OEM Data> ===
```

5932 The version string that follows "SM CLP" is in the [m.n.ud](#) format specified by the *DMTF Release Process*,  
5933 *v1.3* (24HDSP4004) and identifies the version of the SM CLP specification supported by the  
5934 implementation. For implementations compliant with this version of the specification, <major> has a value  
5935 of 1, <minor> has a value of 0 (zero), and <patchlevel> has a value of 0 (zero). The version string that  
5936 follows "SM ME Addressing" is in the [m.n.ud](#) format specified by 24HDSP4004 and identifies the version  
5937 of the SM ME Addressing specification supported by the implementation. <OEM Data> is a vendor-  
5938 defined versioning string. The initial CLP prompt is this version information followed by a command  
5939 prompt

### 5940 **8.2.5 CLP Interaction Model**

5941 The CLP observes the following interaction models documented in this clause, unless otherwise noted:

- 5942 • CLP Service Discovery (documented in the *CLP Service Discovery Specification* [DSP0218])
- 5943 • CLP Session Establishment
- 5944 • CLP Command Interaction
- 5945 • CLP Session Switching
- 5946 • CLP Session Termination

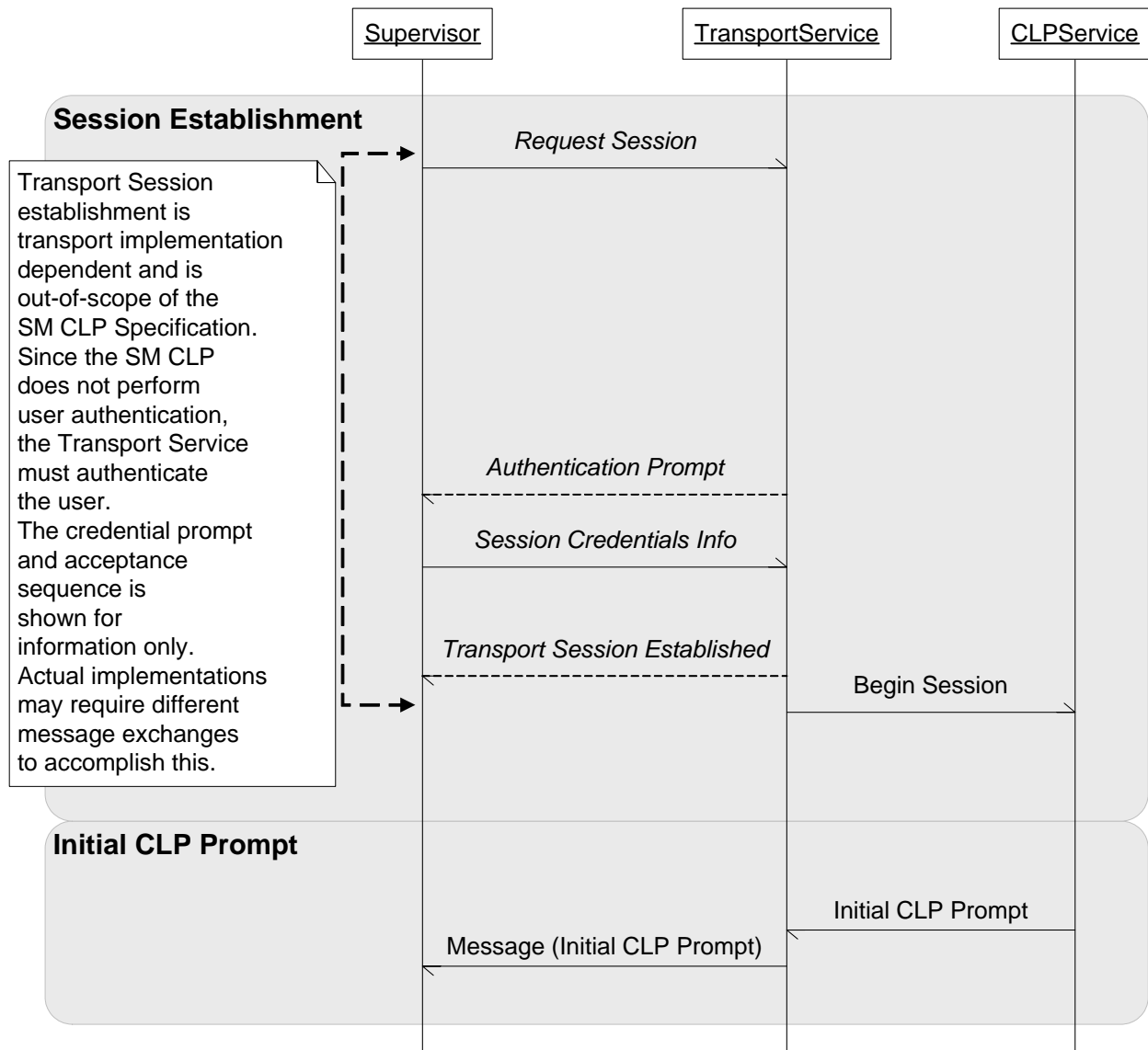
## Server Management Command Line Protocol (SM CLP) Specification

5947 The following clauses define the normal and exception message sequences of each model. The following  
 5948 conventions are used in each of the following interaction diagrams:

- 5949 • Dashed horizontal lines indicate expected behavior. The exact behavior is outside the scope of  
 5950 this specification.
- 5951 • Solid horizontal lines indicate behavior required by this specification.

### 5952 8.2.5.1 Session Establishment

5953 Figure 1 provides an overview of the session establishment sequence.



5954

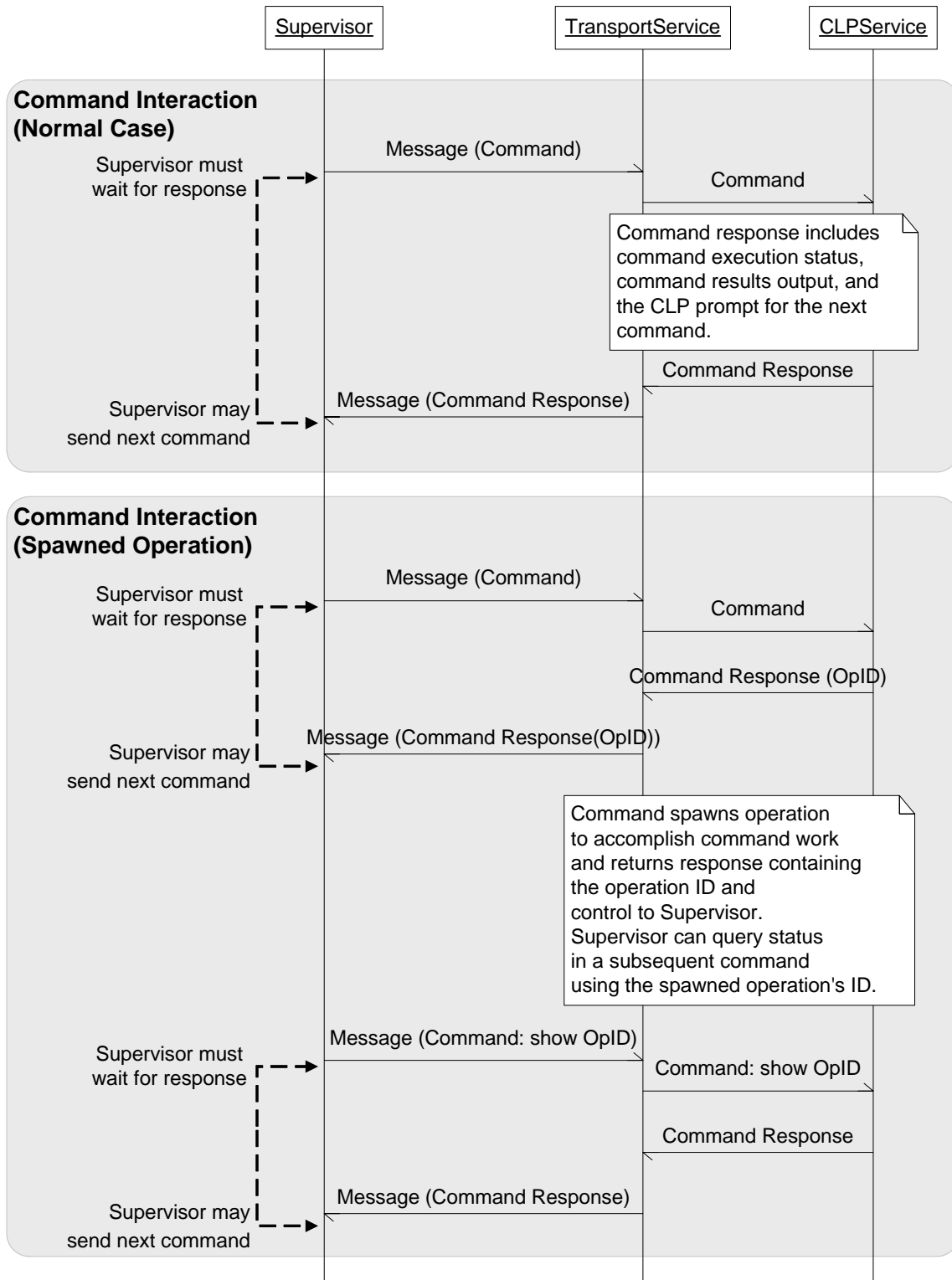
5955 **Figure 1 – Session Establishment Sequence**

### 5956 8.2.5.2 Command Interaction

5957 Figure 2 provides an overview of the interaction between a Supervisor and a CLP Service for a CLP  
 5958 command. Two cases are shown. In the first case, a Command Response is returned synchronously. In



5959 the second case, a command executes asynchronously and the Supervisor polls for status.



5960

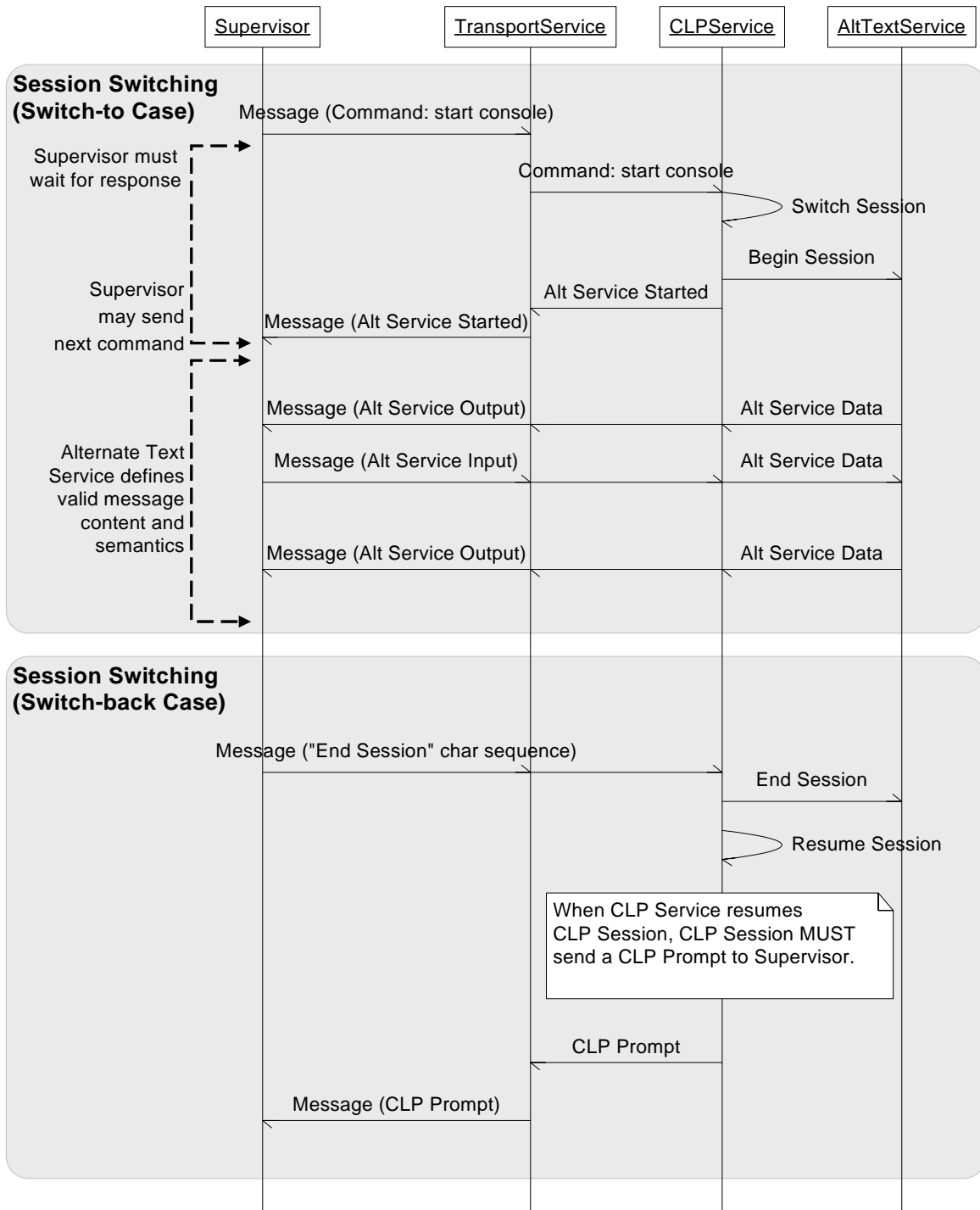
5961

Figure 2 – Command Interaction Sequences

# Server Management Command Line Protocol (SM CLP) Specification

## 5962 8.2.5.3 Session Switching

5963 Figure 3 provides an overview of the interaction between a Supervisor and the CLP Service when an  
 5964 alternate text session is initiated using the CLP. The alternate text session is established within the CLP  
 5965 session. Therefore, while the alternate text session is established, the Supervisor interacts with the  
 5966 alternate text service and not the CLP Service.



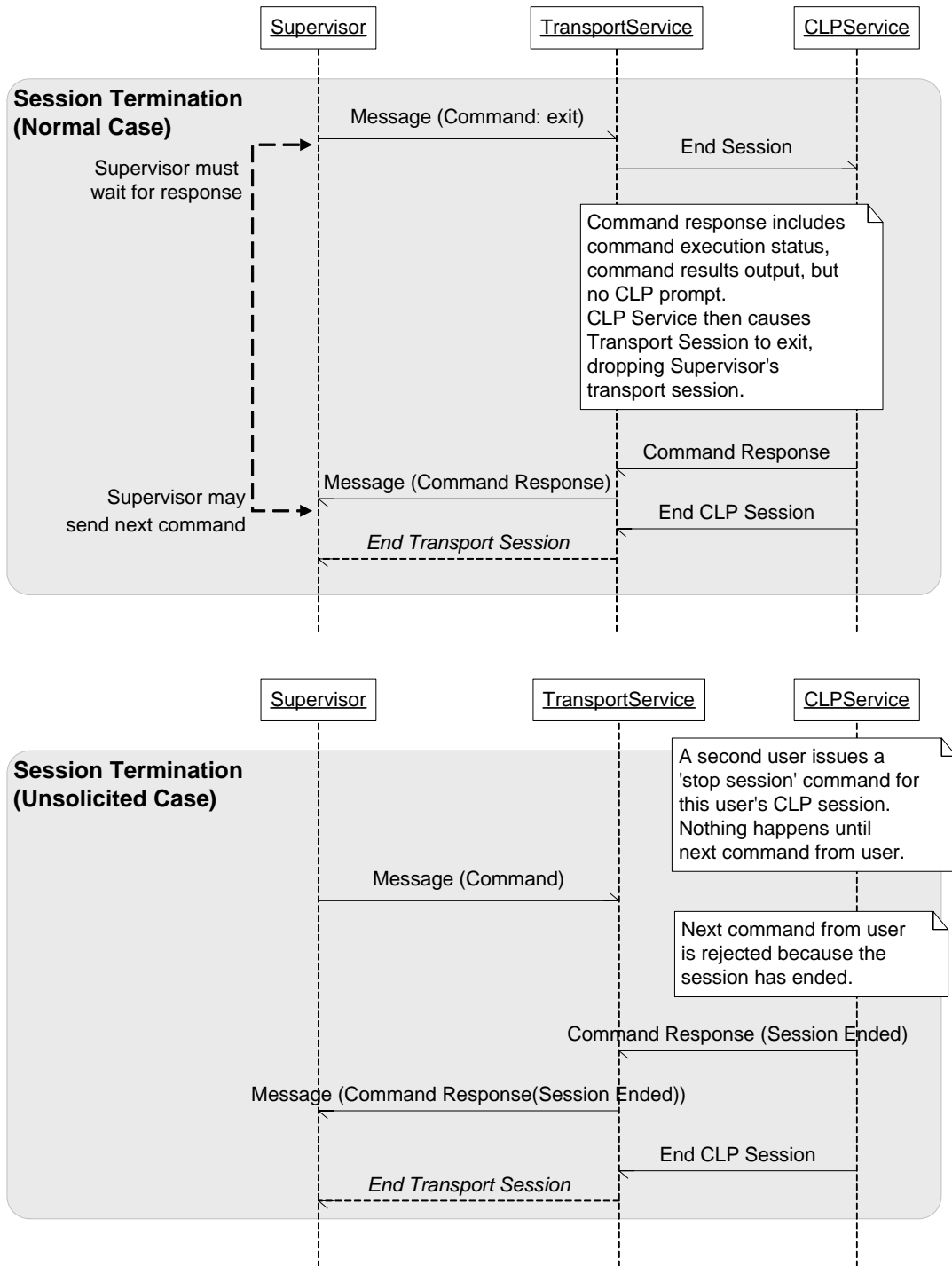
5967

5968

Figure 3 – Session Switching Sequences

5969 **8.2.5.4 Session Termination**

5970 Figure 4 provides an overview of the interaction between a Supervisor and CLP Service when a session  
 5971 terminates. Two cases are shown. In the first case, the Supervisor requests their own session be  
 5972 terminated. In the second case, a different Supervisor requests that the session of the first Supervisor be  
 5973 terminated.



5974

5975

**Figure 4 – Session Termination Sequences**

5976 **8.3 Transport**

5977 The CLP is carried over a text message-based protocol. At present, two text message protocol mappings  
5978 of the CLP are specified:

- 5979 • Telnet
- 5980 • Secure Shell (SSH)

5981 **8.3.1 General**

5982 CLP implementations shall support either Telnet or SSHv2, though support for SSHv2 is recommended.  
5983 Implementations may support more than one message-passing mechanism for accessing the CLP  
5984 Service.

5985 Implementations should provide a method for activating/enabling and deactivating/disabling the supported  
5986 Telnet or SSHv2 protocol.

5987 The CLP Service exposed by the implementation shall operate the same whether invoked from an SSH-  
5988 loaded shell, loaded directly by the SSH daemon, operating as a subsystem, or loaded from a Telnet  
5989 pTTY. This is primarily a test statement because there is nothing in Telnet or SSH that directly affects the  
5990 operation of the implementation or its terminal I/O operations as seen from the transport client application.  
5991 Implementations of the CLP Service shall ensure that output received by Clients of the implementation is  
5992 consistent irrespective of the transport over which the implementation is accessed. For example, if the  
5993 CLP Service uses a transport that inserts characters into the text stream at one end of the transport, the  
5994 transport will need to remove the characters prior to presenting the text stream to the Client.

5995 **8.3.2 Telnet**

5996 Telnet is an IETF-defined Network Virtual Terminal (NVT) protocol that operates over the TCP transport  
5997 layer. The Telnet protocol provides a standardized interface through which a program on one host (the  
5998 Telnet client) can access the resources of another host (the Telnet server) as though the client were a  
5999 local terminal connected to the server.

6000 The basic operational characteristics of an NVT are as follows:

- 6001 • The data representation is 7-bit ASCII transmitted in 8-bit bytes.
- 6002 • The NVT provides a local echo function.

6003 A CLP implementation over Telnet shall support the base Telnet RFC and Standards and shall support  
6004 the IETF RFCs listed in Table 19.

6005 **Table 19 – Telnet Transport Support Requirements**

Number	Title	Author or Ed.	Date Released	Status
STD0008 RFC0854	<i>Telnet Protocol Specification</i>	J. Postel, J.K. Reynolds	May 1, 1983	Standard
STD0008 RFC0855	<i>Telnet Option Specifications</i>	J. Postel, J.K. Reynolds	May 1, 1983	Standard

6006 Note that, in addition to standard NVT operation as described in 22HRFC0854, implementations shall  
6007 respond to requests to support UTF-8.

6008 Each line should be terminated by a Carriage Return and Line Feed (UTF-8, from a *Transformation*  
6009 *Format of ISO 10646* [RFC3629]).

## Server Management Command Line Protocol (SM CLP) Specification

6010 A CLP implementation may listen on any TCP port number. CLP TCP port numbers may be  
6011 administratively set or discovered through the Service Location Protocol (SLP). See 29HDSP0218 for  
6012 more information.

### 6013 8.3.3 Secure Shell (SSH) Version 2

6014 SSH is a tool for secure remote login over insecure networks. It provides an encrypted terminal session  
6015 with strong authentication of both the server and client.

6016 Secure Shell provides three main capabilities:

- 6017 • Secure command-shell
- 6018 • Secure file transfer
- 6019 • Port forwarding

6020 SSHv2 is a protocol being defined by the [IETF](#) in the [SECSH Working Group](#).

6021 The SSHv2 protocol is described in the following five core documents. A CLP implementation over  
6022 SSHv2 shall support the base SSHv2 RFC and Standards and the following IETF RFCs and Internet  
6023 Drafts:

- 6024 • *Secure Shell (SSH) Protocol Architecture* (17HRFC4251)—Describes the overall design of  
6025 SSH.
- 6026 • *Secure Shell (SSH) Transport Layer Protocol* (18HRFC4253)—Provides a single, full-duplex,  
6027 byte-oriented connection between client and server, with privacy, integrity, server  
6028 authentication, and man-in-the-middle protection.
- 6029 • *Secure Shell (SSH) Authentication Protocol* (19HRFC4252)—Identifies the client to the server.
- 6030 • *Secure Shell (SSH) Connection Protocol* (20HRFC4254)—Provides richer, application-support  
6031 services over the transport pipe, such as channel multiplexing, flow control, remote program  
6032 execution, signal propagation, connection forwarding, and so on.
- 6033 • *Secure Shell (SSH) Protocol Assigned Numbers* (RFC4250)—Lists various constant  
6034 assignments made in the other drafts.

6035 The Secure command-shell mode of operation shall be supported by a CLP implementation.

6036 A CLP implementation is not required to listen on any specific predefined TCP port number. A CLP TCP  
6037 port number may be administratively set or discovered through the SLP. See DSP0218 for more  
6038 information.

6039 No specific encryption algorithms or authentication protocols are specified in this specification. CLP will  
6040 rely on the referenced SSH specifications to identify required techniques.

6041 When an implementation supports SSH, the implementation shall support a mechanism to restrict access  
6042 to the implementation using tunneled protocols, and the factory-default setting shall disable Telnet as a  
6043 tunneled protocol.

## Annex A (normative)

### SM CLP Command Grammar in ABNF Notation (RFC2234)

```

6044
6045
6046
6047
6048
6049
6050
6051
6052
6053
6054
6055
6056
6057
6058
6059
6060
6061
6062
6063
6064
6065
6066
6067
6068
6069
6070
6071
6072
6073
6074
6075
6076
6077
6078
6079
6080
6081
6082
6083
6084
6085
6086
6087
6088
6089
6090
6091

```

```

;;
;; Note that line joining is done before tokenization, the "`" (backquote)
character ;; at the end of the line indicates that the current line is continued
the following ;; line.
;;
;; WSP, CR, LF, CRLF, SP, DIGIT, ALPHA, DQUOTE, VCHAR are part of core rules
;; defined in RFC2234.
;;
tsep                = WSP                ; command term separator
eol                 = CR / LF / CRLF     ; end of line token
dot                 = "."
dotdot              = ".."
splat               = "*"
addrTermSeparator  = "/" / "\"         ; address term separator
;;
;; SM CLP command set:
;;
;; General command grammar rules:
;;   verb: always be the 1st term on the command line.
;;   target: appears as the 1st term that is not an option.
;;   options: recognized by the option indication "-", appear after verb.
;;   properties: appear after target.
;;
;;   Example: verb [options] [target] [properties]
;;
Commands = cd-cmd
          / create-cmd
          / delete-cmd
          / dump-cmd
          / exit-cmd
          / help-cmd
          / load-cmd
          / reset-cmd
          / set-cmd
          / show-cmd
          / start-cmd
          / stop-cmd
          / version-cmd
          / OEM-cmd
;;
;; SM CLP command verb set:

```

## Server Management Command Line Protocol (SM CLP) Specification

```
6092 ;;
6093 Verbs = cd-verb
6094     / create-verb
6095     / delete-verb
6096     / dump-verb
6097     / exit-verb
6098     / help-verb
6099     / load-verb
6100     / reset-verb
6101     / set-verb
6102     / show-verb
6103     / start-verb
6104     / stop-verb
6105     / version-verb
6106     / OEM-verb
6107
6108 ;;
6109 ;; SM CLP command options:
6110 ;;
6111 ;; General option grammar rules:
6112 ;;     option = "-" optName [optArg]
6113 ;;     options = option *(tsep option)
6114 ;;
6115 Options = all-option
6116     / destination-option
6117     / display-option
6118     / examine-option
6119     / force-option
6120     / help-option
6121     / keep-option
6122     / level-option
6123     / output-option
6124     / source-option
6125     / version-option
6126     / wait-option
6127     / OEM-options
6128
6129 ;;
6130 ;; SM CLP command properties:
6131 ;;
6132 property = propertyName
6133 propertyArrayName = propertyName "[" 1*DIGIT "]"
6134 properties = 1*(tsep propertyName)
6135 propertyValue = 1*VCHAR / quoted-string ; visible char
6136 propertyValues = [propertyValue] *(", " [propertyValue]) ; for array data type
6137 quoted-string = DQUOTE *(qdtex / ("`" DQUOTE)) DQUOTE
6138 qdtex = %x21 / %x23-7E ; any printable char except DQUOTE
6139 property-assignValue = propertyName "=" propertyValues
6140 property-assignValue = / propertyArrayName "=" propertyValue
6141 property-selectValue = (propertyName / propertyArrayName) "==" propertyValue
```

## Server Management Command Line Protocol (SM CLP) Specification

```
6142 properties-assignValues = 1*(tsep property-assignValue)
6143
6144 ;;
6145 ;; Detail option rule definitions:
6146 ;; Note that <ruleName> denotes an element referenced in this document:
6147 ;; 1. <URI> is defined in RFC2396.
6148 ;; 2. <UFcT>, <UFiT>, and <UFiP> are defined by the SM ME Addressing
6149 Specification.
6150 ;; 3. <propertyName> is defined in Appendix A of CIM Specification V2.2.
6151 ;; 4. <vendor> is a property in CIM_Product class.
6152 ;;
6153 all-option = tsep all-optionName
6154 destination-option = tsep destination-optionName tsep (<URI> / <UFiP>)
6155 display-option = tsep display-optionName tsep display-optionArgs
6156 *("," display- optionArgs)
6157 display-optionPropArgProperty = (propertyName)
6158 display-optionArgs = *( "all" / "verbs" / display-optionAssocArg /
6159 display-optionTargetsArg / display-optionPropArg )
6160 display-optionAssocArg = "associations" ["=" (className /
6161 ("("className *("," className))"")) ]
6162 display-optionTargetsArg = "targets" ["=" (<UFcT> /
6163 ("("<UFcT> *("," <UFcT>)"")) ]
6164 display-optionPropArg = "properties" ["=" display-optionPropArgProperty /
6165 ( "(" display-optionPropArgProperty *("," display-optionPropArgProperty ) )" ) ]
6166 examine-option = tsep examine-optionName
6167 force-option = tsep force-optionName
6168 help-option = tsep help-optionName
6169 keep-option = tsep keep-optionName tsep 1*DIGIT [ "." 1*DIGIT ]
6170 level-option = tsep level-optionName tsep (1*DIGIT / "all")
6171 output-option = tsep output-optionName tsep output-optionArgs
6172 *("," output-optionArgs)
6173 output-optionFormatArg = ("text" / "keyword" / "clpxml")
6174 output-optionArgs = *( ("format" "=" output-optionFormatArg /
6175 ( "(" output-optionFormatArg )" ) )
6176 / ("error" / "terse" / "verbose")
6177 / ("language" "=" 3ALPHA / ( "(" 3ALPHA )" ) )
6178 / ("begin" / "end")
6179 / ("order" "=" ("default" / "reverse") / ( "(" "default" / "reverse" )" ) )
6180 / ("number" "=" (1*DIGIT "-" 1*DIGIT) / ( "(" (1*DIGIT "-" 1*DIGIT) )" ) )
6181 / ("count" "=" ("all" / 1*DIGIT) / ( "(" ("all" / 1*DIGIT) )" ) )
6182
6183 source-option = tsep source-optionName tsep (<URI> / <UFiP>)
6184 version-option = tsep version-optionName
6185 wait-option = tsep wait-optionName
6186 stateValue = propertyValue
6187
6188 ;;
6189 ;; Common options applicable to all verbs:
6190 ;;
6191 common-options = *(examine-option / help-option / keep-option / output-options /
6192 version-option / OEM-options)
6193
```



## Server Management Command Line Protocol (SM CLP) Specification

```
6194 ;;
6195 ;; Command target term formations:
6196 ;;
6197 all-legal-targets = target-Assoc / target-Instance / target-UFsT
6198 target-Instance = tsep targetPath
6199 / tsep OEM-target
6200 / tsep "SESSION"
6201 target-UfsT = tsep [addrTermSeparator] [addrTerm *(addrTermSeparator addrTerm)
6202 addrTermSeparator] UFST
6203 UFST = <UFcT> splat ; UFcT*
6204 target-Assoc = target-AssocSingleInstance / target-AssocMultiInstance
6205 target-AssocSingleInstance = tsep targetPath "=>" className "=>" targetPath
6206 target-AssocMultiInstance = tsep targetPath "=>" className
6207 targetPath = [addrTermSeparator] [addrTerm *(addrTermSeparator addrTerm)]
6208 addrTerm = (dot / dotdot / <UFiT>)
6209
6210 ;;
6211 ;; Top-level command line production
6212 ;;
6213 clp-command-line = clp-verb-forms / oem-cmd
6214
6215 ;;
6216 ;; The CLP command formations
6217 ;;
6218 clp-verb-forms = cd-cmd / create-cmd / delete-cmd / dump-cmd / exit-cmd
6219 clp-verb-forms =/ help-cmd / load-cmd / reset-cmd / set-cmd / show-cmd
6220 clp-verb-forms =/ start-cmd / stop-cmd / version-cmd
6221 ;;
6222 ;; Per-command formations:
6223 ;;
6224 cd-cmd = cd-verb [cd-options] [target-Instance] eol
6225 cd-options = *(wait-option / common-options)
6226
6227 create-cmd = create-verb [create-options] (target-Instance / target-UFST)
6228 properties-assignValues eol
6229 create-cmd =/ create-verb [create-options] source-option [create-options]
6230 (target-Instance / target-UFST) eol
6231 create-cmd =/ create-verb (help-option / version-option) eol
6232 create-options = *(wait-option / common-options)
6233
6234 delete-cmd = delete-verb [delete-options] [target-Instance / target-UFST] eol
6235 delete-options = *(force-option / wait-option / common-options)
6236
6237 dump-cmd = dump-verb [dump-options] destination-option [dump-options] [target-
6238 Instance] eol
6239 dump-cmd =/ dump-verb (help-option / version-option) eol
6240 dump-options = *(force-option / wait-option / common-options)
6241
6242 exit-cmd = exit-verb [exit-options] eol
6243 exit-options = common-options
6244
```

## Server Management Command Line Protocol (SM CLP) Specification

```
6245 help-cmd = help-verb [help-options] [1*VCHAR *( tsep 1*VCHAR)] eol
6246 help-options = *(wait-option / common-options)
6247
6248 load-cmd = load-verb [load-options] source-option [load-options] [target-Instance]
6249 eol
6250 load-cmd =/ load-verb (help-option / version-option) eol
6251 load-options = *(force-option / wait-option / common-options)
6252
6253 reset-cmd = reset-verb [reset-options] [target-Instance] eol
6254 reset-options = *(force-option / wait-option / common-options)
6255
6256 set-cmd = set-verb [set-options] [target-Instance / target-AssocSingleInstance]
6257 properties-assignValues eol
6258 set-options = *(force-option / wait-option / common-options)
6259
6260 show-cmd = show-verb [level-option] [show-options] [level-option] [target-
6261 Instance] *(propertyName / property-selectValue) eol
6262 show-cmd =/ show-verb [show-options] [target-UFsT / target-Assoc] *(propertyName /
6263 property-selectValue) eol
6264 show-options = *(all-option / display-option / force-option / wait-option /
6265 common-options)
6266
6267 start-cmd = start-verb [start-options] [target-Instance] eol
6268 start-options = *(force-option / wait-option / common-options)
6269
6270 stop-cmd = stop-verb [stop-options] [target-Instance] eol
6271 stop-options = *(force-option / wait-option / common-options)
6272
6273 version-cmd = version-verb [version-options] eol
6274 version-options = *(wait-option / common-options)
6275
6276 ;;
6277 ;; OEM syntax:
6278 ;;
6279 OEM-cmd = "OEM"<vendor><vendor-specified command line syntax> eol
6280 OEM-verb = "OEM"<vendor><vendor-specified verb name>
6281 OEM-target = tsep "OEM"<vendor><vendor-specified targetAddress>
6282 OEM-options = tsep *(OEM-optionName [tsep OEM-optionArgs])
6283 OEM-optionName = "-OEM"<vendor><vendor-specified option Name>
6284 OEM-optionArgs = "OEM"<vendor><vendor-specified option arguments>
6285 OEM-propertyName = tsep "OEM"<vendor><vendor-specified property name>
6286
6287 ;;
6288 ;; Verb names:
6289 ;;
6290 cd-verb = "cd" ; Note that ABNF strings are case-insensitive
6291 create-verb = "create"
6292 delete-verb = "delete"
6293 dump-verb = "dump"
6294 exit-verb = "exit"
6295 help-verb = "help"
6296 load-verb = "load"
```

## Server Management Command Line Protocol (SM CLP) Specification

```
6297 reset-verb      = "reset"
6298 set-verb        = "set"
6299 show-verb       = "show"
6300 start-verb      = "start"
6301 stop-verb       = "stop"
6302 version-verb    = "version"
6303
6304 ;;
6305 ;; Option names:
6306 ;;
6307 all-optionName  = "-a" ["ll"]
6308 destination-optionName = "-destination"
6309 display-optionName = "-d" ["isplay"]
6310 examine-optionName = "-x" / "-examine"
6311 force-optionName = "-f" ["orce"]
6312 help-optionName = "-h" ["elp"]
6313 keep-optionName = "-k" ["eep"]
6314 level-optionName = "-l" ["evel"]
6315 output-optionName = "-o" ["utput"]
6316 source-optionName = "-source"
6317 version-optionName = "-v" ["ersion"]
6318 wait-optionName = "-w" ["ait"]
6319
```

**Annex B  
(informative)**

6320  
6321  
6322  
6323  
6324

**W3C Universal Resource Identifiers (URI)**

6325 URIs are expected to be used as values for some keyword=value pairs, option arguments, and option  
6326 argument values. For instance, an implementation may use a URI as a boot device as the location of the  
6327 data source for applying a firmware image. The implementation is expected to validate the URI and  
6328 ensure that the schema name included in the URI is valid for the given implementation. The CLP itself  
6329 does not require any specific schema or enforce any specific URIs, but they are expected to adhere to  
6330 RFC2396, RFC2718, and RFC2717.

**Annex C  
(informative)**

6331  
6332  
6333  
6334  
6335

**W3C Extensible Markup Language (XML)**

6336 The CLP supports generating XML output data (*Extensible Markup Language [XML] 1.0, 3<sup>d</sup> edition*), as  
6337 well as keyword mode and modes for plain text output. XML was chosen as a supported output format  
6338 due to its acceptance in the industry, establishment as a standard, and the need for Clients to import data  
6339 obtained through the CLP into other applications. For more information on the XML output mode, see  
6340 5.2.4.3.3.

6341 **Annex D**  
6342 **(informative)**

6343  
6344 **POSIX Utility Conventions**  
6345

6346 The POSIX Utility Conventions (IEEE Std. 1003.1) were considered when defining the CLP syntax. The  
6347 CLP syntax adheres to as many of the POSIX Utility Guidelines as are feasible, but it does not conform to  
6348 the POSIX Utility Argument Syntax.

6349 The POSIX Utility Argument Syntax was found inappropriate for two reasons. First, it was imperative to  
6350 have the command target term be deterministic in order to accommodate low-end implementations.  
6351 Second, in order to provide a consistent, predictable mapping to the CIM Schema, the CLP syntax uses  
6352 the convention that option terms apply to command verbs and parameter terms apply to the command  
6353 target, using the "keyword=value" model.

6354 The CLP syntax compares to the thirteen POSIX Utility Guidelines as follows:

6355 Adhering to Guideline 1 is a goal of the CLP, because it is desirable to keep the verb names short.  
6356 However, the adopted extensibility conventions imply that it is expected that any extensions will find it  
6357 problematic to adhere to Guideline 1.

6358 The CLP syntax currently has no numbers in the commands, nor are verbs required to be entered only in  
6359 lowercase. Therefore, a user or script that adheres to Guideline 2 could find CLP implementations  
6360 compatible.

6361 The CLP allows both a short name form and long name form for option names. Therefore any human  
6362 user or script that is accustomed to one-letter option names, as established in Guideline 3, will find CLP  
6363 implementations compatible. Allowing whole word options not only allows scripts to be more readable, but  
6364 allows a shorter learning curve of the CLP. The "W" option is not reserved by the CLP. CLP option names  
6365 are case insensitive.

6366 The CLP adheres to Guideline 4: all CLP options are preceded by the '-' (hyphen) delimiter character.

6367 The CLP does not allow grouping of options behind a single hyphen and therefore does not adhere to  
6368 Guideline 5. Most options require a parameter, and the decision to allow full-length option names  
6369 eliminated the ability to adhere to this guideline.

6370 The CLP adheres to Guideline 6 and recognizes the space character as the command line term delimiter.

6371 The CLP adheres to Guideline 7: each option either always requires an argument or never requires an  
6372 argument.

6373 The CLP recognizes the use of the comma character to separate items in a list in a single argument string  
6374 for both options and properties and therefore adheres to Guideline 8 with one caveat. A comma character  
6375 at the beginning or end of the option argument string is not inherently illegal and is command dependent.

6376 The CLP adheres to Guideline 9: the command line form is in the order of Verb, Option, Target, Property.

6377 The CLP does not recognize the "--" (hyphen hyphen) term as an "end of options" indicator, nor as a  
6378 "long option" indicator (as is used in some UNIX utilities). Therefore, the CLP does not adhere to  
6379 Guideline 10.

6380 The CLP allows options to be specified in any order, but it does not allow options to appear twice on any  
6381 command line, nor does it allow mutually exclusive options or options that do not apply in the current  
6382 context. Therefore, the CLP adheres to part of Guideline 11.

## Server Management Command Line Protocol (SM CLP) Specification

- 6383 When examining Guideline 12, the CLP uses keyword=value pairs for operands that require assignment,  
6384 and just keywords for operands that do not. Because these are often CIM Schema properties, they are  
6385 not order dependent. Therefore, the positions of operands do not matter. This is true regardless of the  
6386 CLP command.
- 6387 Guideline 13 is out of scope for the CLP. The CLP does not allow in-stream input and therefore has no  
6388 need for an input operand.

6389 **Annex E**  
6390 **(informative)**

6391  
6392  
6393 **Conventions**

6394 The terms "implicit" and "default" are used in this document to describe aspects of the protocol as follows:

6395 Functions or behaviors are defined to be "implicit" if those functions or behaviors are an integral part of  
6396 the protocol definition and cannot be overridden by command or command options.

6397 Functions or behaviors are defined to be "default" if those functions or behaviors are assumed to be in  
6398 effect unless overridden or specified by the user through a command or command option.

6399



6400 **Annex F**  
6401 **(informative)**

6402  
6403  
6404 **Notation**

6405 Regular Expression (regex) and Augmented Backus-Naur Form (ABNF) are used in this document to  
6406 describe various aspects of the SM CLP specification. A complete SM CLP grammar in ABNF is in  
6407 Annex A.

6408 For readability, this specification documents all verb, option, target, and property names in lowercase.

6409 When command option names have multiple, supported forms, each form is listed explicitly, separated by  
6410 a comma. For example, the `level` command option has two acceptable forms: `-l` and `-level`. The  
6411 specification text lists these alternatives as `-l, -level`.

6412 The following conventions are used to indicate specification elements:

6413 `courier new` Used to indicate literal characters in the syntax expression and in examples.

6414 *italicized* Used to indicate the type or description of data to be inserted.

6415 `< >` Used to indicate terms in an expression.

6416 Capitalization Used to indicate defined terms.

6417 Examples are provided for informative purposes and are shown using Courier New font. When the text  
6418 provides an example of a CLP command and response, the CLP Command Line is emphasized using  
6419 bold text. The command output is shown in regular text font.

6420 Examples are for information only. When an example contradicts specification text elsewhere in the  
6421 document, the specification text is the authority. Examples are shown `in this font and format`. Each  
6422 example consists of a description of the example, the CLP Command Line emphasized using bold text,  
6423 and the Command Response in flat text. General rules and requirements for the Command Response are  
6424 specified in 5.1.10. When examples do not include the `output` option with a `format` argument, it is  
6425 assumed that session default format is that of the example.

6426

**Annex G**  
**(informative)**

**Change History**

6427  
6428  
6429  
6430  
6431

Version 1.02	2007-02-19	Formatted to conform to <i>ISO/IEC Directives, Part 2: Rules for the structure and drafting of International Standards</i>
--------------	------------	--

6432 **Annex H**  
6433 **(informative)**

6434  
6435  
6436 **Acknowledgements**

6437 The authors wish to acknowledge the following people.

6438 Editors:

- 6439 • Aaron Merkin, IBM  
6440 • Perry Vincent, Intel

6441 Contributors:

- 6442 • Bob Blair, Newisys  
6443 • Greg Dake, IBM  
6444 • Jon Hass, Dell  
6445 • Jeff Hilland, HP  
6446 • Steffen Hulegaard, OSA Technologies  
6447 • Arvind Kumar, Intel  
6448 • Jeff Lynch, IBM  
6449 • Christina Shaw, HP  
6450 • Enoch Suen, Dell  
6451 • Michael Tehranian, Sun

6452

6453 **Annex I**  
6454 **(informative)**

6455  
6456  
6457 **Bibliography**

- 6458 [IEEE Std 1003.1](#), "POSIX Utility Conventions", *The Open Group Base Specifications Issue 6*, 2004  
6459 Edition
- 6460 World Wide Web Consortium (W3C), [Extensible Markup Language \(XML\) 1.0 \(Third Edition\)](#), February  
6461 2004
- 6462 IETF, [RFC4251](#), *Secure Shell (SSH) Protocol Architecture*, January 2006
- 6463 IETF, [RFC4253](#), *Secure Shell (SSH) Transport Layer Protocol*, January 2006
- 6464 IETF, [RFC4252](#), *Secure Shell (SSH) Authentication Protocol*, January 2006
- 6465 IETF, [RFC4254](#), *Secure Shell (SSH) Connection Protocol*, January 2006
- 6466 IETF, [RFC4250](#), *Secure Shell (SSH) Protocol Assigned Numbers*, January 2006
- 6467 IETF, [RFC0854](#), *Telnet Protocol Specification*, May 1983
- 6468 IETF, [RFC0855](#), *Telnet Option Specifications*, May 1983
- 6469 DMTF, [DSP4004](#), *DMTF Release Process*, version 1.6.0, January 2007
- 6470 OMG, [Unified Modeling Language \(UML\) from the Open Management Group \(OMG\)](#)
- 6471 DMTF, [DSP2001](#), *Systems Management Architecture for Server Hardware (SMASH) Command Line  
6472 Protocol (CLP) Architecture White Paper*, version 1.0.1, October 2006
- 6473 IETF, [RFC2718](#), *Guidelines for new URL Schemes*, November 1999
- 6474 IETF, [RFC2717](#), *Registration Procedures for URL Scheme Names*, November 1999
- 6475 DMTF, [DSP0218](#), *Server Management (SM) Command Line Protocol (CLP) Discovery Using the Service  
6476 Location Protocol (SLP)*, version 1.0, 2005
- 6477 IETF, [RFC2821](#), *Simple Mail Transfer Protocol*, April 2001
- 6478 DMTF, [DSP0207](#), *Web-Based Enterprise Management (WBEM) Universal Resource Identifier (URI)  
6479 Mapping Specification*, version 1.0, January 2006
- 6480 DMTF, [DSP0216](#), *Command Line Protocol (CLP)-to-Common Information Model (CIM) Mapping  
6481 Specification*, version 1.0, 2005
- 6482 DMTF, [DSP0217](#), *Systems Management Architecture for Server Hardware (SMASH) Implementation  
6483 Requirements*, version 1.0, 2005
- 6484 IETF, [RFC3629](#), *UTF-8, a transformation format of ISO 10646*, 2003  
6485