



1  
2  
3  
4

**Document Number: DSP0215**

**Date: 2009-04-23**

**Version: 1.0.0**

5 **Server Management Managed Element**  
6 **Addressing Specification**

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

10

11 Copyright notice

12 Copyright © 2005, 2009 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

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

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27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party  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

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

33

34

35

## CONTENTS

36	Foreword .....	5
37	Introduction .....	6
38	1 Scope .....	8
39	2 Normative References.....	8
40	2.1 Approved References .....	8
41	2.2 Other References.....	8
42	3 Terms and Definitions .....	8
43	4 User Friendly Class Tags.....	13
44	5 SM Containment Hierarchy .....	23
45	5.1 General Concepts .....	23
46	5.2 Logical Containment .....	23
47	5.3 Physical Containment .....	25
48	5.4 SM ME Addressing Hierarchy.....	29
49	5.5 Addressing Associations.....	32
50	6 SM ME Addressing.....	33
51	6.1 Instance Tagging .....	33
52	6.2 Addressing Rules.....	33
53	6.3 Addressing Grammar.....	34
54	ANNEX A (informative) Considerations for Implementation .....	47
55	ANNEX B (informative) Document Conventions.....	49
56	B.1 CIM_Prefix .....	49
57	B.2 Notation.....	49
58	ANNEX C (informative) Change Log .....	50
59	Bibliography .....	51
60		

### 61 Figures

62	Figure 1 – SM Logical Containment UML Diagram .....	24
63	Figure 2 – SM Physical Containment UML Diagram .....	26
64	Figure 3 – CIM PhysicalElement Class Hierarchy .....	27
65	Figure 4 – High-Level View of the SM ME Addressing Hierarchy .....	30

66

### 67 Tables

68	Table 1 – UFcTs for Subclasses of CIM_ManagedElement.....	13
69	Table 2 – UFcTs for Subclasses of CIM_Capabilities .....	14
70	Table 3 – UFcTs for Subclasses of CIM_SettingData .....	14
71	Table 4 – UFcTs for Subclasses of CIM_Collection .....	15
72	Table 5 – UFcTs for Subclasses of CIM_LogicalElement .....	15
73	Table 6 – UFcTs for Subclasses of CIM_EnabledLogicalElement .....	15
74	Table 7 – UFcTs for Subclasses of CIM_System .....	16
75	Table 8 – UFcTs for Subclasses of CIM_LogicalDevice.....	17
76	Table 9 – UFcTs for Subclasses of CIM_Service .....	19
77	Table 10 – UFcTs for Subclasses of CIM_ServiceAccessPoint .....	19

78	Table 11 – UFcTs for Subclasses of CIM_PhysicalElement .....	20
79	Table 12 – SM Hardware Containment Relationships .....	31
80	Table 13 – SM Logical Containment Relationships .....	31
81	Table 14 – SM ME Addressing Rules .....	33
82	Table A-1 – Examples of SM UFITs .....	47

83

84

## Foreword

85 The *Server Management Managed Element Addressing Specification* (DSP0215) was prepared by the  
86 Server Management Working Group of the DMTF.

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

### 89 **Acknowledgments**

90 The authors wish to acknowledge the following contributors and participants from the DMTF Server  
91 Management Working Group:

- 92 • Aaron Merkin – IBM
- 93 • Jeff Hilland – HP
- 94 • Jon Hass – Dell
- 95 • Khachatur Papanyan – Dell
- 96 • Radhakrishna Dasari – Dell
- 97 • Jeff Hilland – HP
- 98 • Christina Shaw – HP
- 99 • John Leung – Intel

100

101

## Introduction

102 This document describes the Server Management (SM) Managed Element (ME) addressing standard. SM  
103 ME addressing provides an easy, user friendly way to address Common Information Model (CIM) objects  
104 (classes and instances). This specification may be used to define valid targets for [SM CLP commands](#).

### 105 Overview

106 CIM is a rich modeling environment in that it provides many ways to model computing environments. It  
107 allows different vendors to choose different model components to model their computing equipment and  
108 environments. While this open-ended modeling behavior is desirable in the theoretical sense, it poses  
109 difficulties in a practical environment in that it often creates a situation in which different vendors' tools  
110 cannot interoperate because they have different notions of the computing model. SM ME addressing  
111 provides an approach to avoid these difficulties.

112 SM ME addressing is based on the CIM containment hierarchy described in section 5. The containment  
113 hierarchy defines the hierarchical CIM container-based addressing associations that are combined to  
114 form a fully qualified address path to an instance, much like the path to a file in a file system. The  
115 containment hierarchy, which is based on relevant ME classes and associations, forms the basis of the  
116 addressing grammar described in section 6. The SM profile specifications define the specific CIM objects  
117 that may be addressed and their supported non-addressing associations. The [SM CLP Specification](#)  
118 defines how to use an SM ME address in a CLP command and how to discover what non-addressing  
119 associations are supported for any given target ME.

120 A fully qualified address path (called a UFIP) to a User Friendly instance Tag (UFIT) addresses a single  
121 CLP command target and may be mapped to a specific CIM object reference, thus providing support for  
122 communication between the Manageability Access Point (MAP) and CIM servers, clients, and providers.  
123 These features combine to enable an embedded lightweight CIM server in the Manageability Access  
124 Point.

125 Several key components form the technical foundation that enables SM ME addressing to provide a  
126 simple, user friendly, unique address path to a CIM instance. These key components are as follows:

- 127 • SM profiles
- 128 • SM ME grammar
- 129 • CIM model

130 The following sections discuss the role that each of these key technologies plays in the SM addressing  
131 scheme.

### 132 SM Profiles

133 The SM containment hierarchy defined in section 5 forms the basis for SM ME addressing by identifying  
134 the hierarchical containment associations between objects. The SM profiles define the model components  
135 that implementations must support. The SM ME addressing rules define how implementations create  
136 UFITs for instances of classes specified in the SM profiles.

137 SM profiles define rules and guidelines to model specific hardware platforms such as base systems,  
138 modular (blade) systems, and other architectural features such as clustering, redundancy (resource  
139 sharing), and virtual machines. Other SM profiles define rules and guidelines to model specific  
140 management domains such as Boot Control, Power Control, Firmware Update, and the like. This  
141 addressing specification has the grammar and rules associated with the SM profiles.

142 If implementations follow the models specified in the profiles and the addressing rules, then fully qualified  
143 UFiPs are guaranteed to exist for all managed instances in the implementations' namespace. The SM  
144 profiles guarantee that all conforming implementations provide the same MEs and supporting  
145 associations. The SM containment hierarchy guarantees that these MEs are addressable.

#### 146 **SM ME Grammar**

147 The SM ME addressing grammar described in section 6.3 provides unambiguous productions that  
148 guarantee that a programmatic parser can be coded to detect valid ME addresses. Furthermore, it  
149 guarantees that each valid production eventually terminates in a unique UFiT or User Friendly class Tag  
150 (UFcT). The difference between the UFcT and the UFiT is the unique instance suffix, which identifies a  
151 particular CIM instance of a class.

#### 152 **CIM Model**

153 The CIM schema defines the valid associations between CIM instances. This specification provides the  
154 rules for subsets of CIM associations called addressing associations, which are used to form ME  
155 addresses. These rules are used to validate terms in a UFiP. The UFiTs on each side of the association  
156 path must have the proper object type and role. CIM defines the properties for each object class and its  
157 subclasses.

158 CIM was chosen as the underlying data model because of its capacity to normalize computer-based  
159 management relationships and information. The CIM schema describes a well-defined structured  
160 containment hierarchy that can be adapted to provide an unambiguous method of addressing MEs.

#### 161 **Target Audience**

162 This specification is intended to guide developers of SM ME addressing implementations. It may also be  
163 used as a reference by system administrators and other users of an SM ME addressing implementation.

#### 164 **SM ME Addressing Goals**

165 The goals of the *Server Management Managed Element Addressing Specification* are as follows:

- 166 • Provide a user friendly way to accurately address CIM objects, using a hierarchical containment  
167 structure based on specified CIM associations. This addressing mechanism is intended to be  
168 applicable to other DMTF protocols.
- 169 • Provide access to information in other Managed Element instances associated with the target  
170 ME instance through non-addressing associations that are not part of a SM containment  
171 hierarchy. This feature is required to support  $n$  dimensional association traversal rooted at the  
172 terminating target UFiT.  $n$  is bounded by the non-addressing associations defined in the SM  
173 profiles supported by the MAP. Addressing associations are defined in 5.4.
- 174 • Provide an unambiguous grammar to aid in programmatic parsing of a UFiP. The grammar is  
175 defined in 6.3.

176

# 177 Server Management Managed Element Addressing Specification

## 178 1 Scope

179 This document describes the Server Management (SM) Managed Element (ME) addressing standard. SM  
180 ME addressing provides an easy, user friendly way to address Common Information Model (CIM) objects  
181 (classes and instances). This specification may be used to define valid targets for [SM CLP commands](#).

## 182 2 Normative References

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

### 186 2.1 Approved References

187 DMTF, *Common Information Model (CIM) Schema*, 2.10, <http://www.dmtf.org/standards/cim>

188 DMTF DSP0214, *SM Command Line Protocol Specification*, 1.0.0,  
189 [http://www.dmtf.org/standards/published\\_documents/DSP0214.pdf](http://www.dmtf.org/standards/published_documents/DSP0214.pdf)

190 IETF RFC 2234, *Augmented BNF for Syntax Specifications: ABNF*, November 1997,  
191 <http://www.ietf.org/rfc/rfc2234.txt>

192 OMG, *Unified Modeling Language (UML) from the Open Management Group (OMG)*, <http://www.uml.org/>

### 193 2.2 Other References

194 ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*  
195 <http://isotc.iso.org/livelink/livelink?func=ll&objId=4230456&objAction=browse&sort=subtype>  
196

## 197 3 Terms and Definitions

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

### 199 3.1

#### 200 **can**

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

### 202 3.2

#### 203 **cannot**

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

### 205 3.3

#### 206 **conditional**

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



- 209 **3.4**  
210 **mandatory**  
211 indicates requirements to be followed strictly in order to conform to the document and from which no  
212 deviation is permitted
- 213 **3.5**  
214 **may**  
215 indicates a course of action permissible within the limits of the document
- 216 **3.6**  
217 **need not**  
218 indicates a course of action permissible within the limits of the document
- 219 **3.7**  
220 **optional**  
221 indicates a course of action permissible within the limits of the document
- 222 **3.8**  
223 **shall**  
224 indicates requirements to be followed strictly in order to conform to the document and from which no  
225 deviation is permitted
- 226 **3.9**  
227 **shall not**  
228 indicates requirements to be followed in order to conform to the document and from which no deviation is  
229 permitted
- 230 **3.10**  
231 **should**  
232 indicates that among several possibilities, one is recommended as particularly suitable, without  
233 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 234 **3.11**  
235 **should not**  
236 indicates that a certain possibility or course of action is deprecated but not prohibited
- 237 **3.12**  
238 **Addressing Association**  
239 a certain CIM association that is valid for the purpose of addressing CIM instances defined in SM profiles  
240 These CIM associations are based on the SM containment hierarchy. In the SM addressing grammar this  
241 association is denoted by a forward slash (/) or a backward slash (\) in the ME's UFiP. These  
242 associations are enumerated in Table 12 and Table 13, which define the logical and physical address  
243 containment.
- 244 **3.13**  
245 **Address Path**  
246 a path in which each term has the appropriate intervening addressing association  
247 This is explained in detail in 5.4 and 6.2.
- 248 **3.14**  
249 **client**  
250 a logical component that manages a system through a Manageability Access Point ([MAP](#))  
251 A client may sometimes be referred to as the managing entity or management client.

- 252 **3.15**  
253 **instance suffix**  
254 a positive non-zero integer appended to the end of a [UFcT](#) to create an instance tag or [UFiT](#)  
255 The instance suffix creates a unique identifier for the instance within the defining container. A UFcT plus  
256 an instance suffix is a UFiT.
- 257 **3.16**  
258 **Logical and Physical Containment**  
259 a concept that an object can contain another object or be contained within another object  
260 The SM ME addressing scheme is based on the notion of object containment. In the logical CIM schema  
261 the ComputerSystem is the container for objects such as CIM Service, ServiceAccessPoint,  
262 LogicalDevice, and SystemSpecificCollection. In the physical CIM schema the Rack is the top most  
263 physical container. Table 12 and Table 13 describe the containers, the CIM relationship, and the  
264 instances they contain.
- 265 **3.17**  
266 **Manageability Access Point**  
267 **MAP**  
268 a network-accessible interface for managing a Computer System  
269 A MAP can be instantiated by a Management Process, a Management Processor, a Service Processor,  
270 or a Service Process.
- 271 **3.18**  
272 **Managed Element**  
273 **ME**  
274 an instance of CIM\_ManagedElement
- 275 **3.19**  
276 **Managed Element Address**  
277 the [UFiP](#) to an ME instance  
278 The UFiP may be used as a target of a SM CLP command.
- 279 **3.20**  
280 **Managed System Address Space**  
281 the fully connected graph of [UFiTs](#) that are contained in a specific managed system  
282 The root instance is the ComputerSystem instance that represents the managed system.
- 283 **3.21**  
284 **MAP Address Space**  
285 the hierarchical graph of the [UFiTs](#) contained in the MAP's AdminDomain  
286 Each instance starting at the AdminDomain is a node in the graph. Each supported association forms a  
287 link in the graph to another instance node and so on until a terminating instance node is encountered.
- 288 **3.22**  
289 **Non-Addressing Association**  
290 a CIM association class that is used in an SM profile but is not one of the SM addressing associations  
291 that are denoted by a forward slash (/) or a backward slash (\) in the [UFiP](#). See 6.3.
- 292 **3.23**  
293 **Physical Element**  
294 **PE**  
295 a tangible element that has a physical manifestation of some sort (as defined in CIM)

- 296 **3.24**  
297 **Root Instance**  
298 the top node in a SM address space; the MAP's AdminDomain
- 299 **3.25**  
300 **Server Management**  
301 **SM**  
302 the ability to remotely administer server-class computer platforms, such as rack-mounted, stand-alone,  
303 and blade servers and their respective enclosures
- 304 **3.26**  
305 **System Management Architecture for Server Hardware**  
306 **SMASH**  
307 an initiative to codify the management interface for heterogeneous servers in the data center,  
308 independent of machine state, operating system state, system topology or access method
- 309 **3.27**  
310 **SM Command Line Protocol**  
311 **SM CLP**  
312 a user friendly command-line protocol to manipulate CIM instances defined by the SM profiles
- 313 **3.28**  
314 **SM profile**  
315 a profile defined or referenced by an SM profile specification  
316 These profile specifications define CIM profiles for required and recommended instances, properties, and  
317 expected behavior to consistently model computer hardware platforms and management domains in  
318 order to achieve implementation interoperability. Many of these profiles are listed in [DSP0217](#).
- 319 **3.29**  
320 **Standard User Friendly class Tags**  
321 the User Friendly class Tags listed in the UFcT column of Table 1 through Table 11
- 322 **3.30**  
323 **User Friendly class Tag**  
324 **UFcT**  
325 a short, user friendly synonym for a CIM class name  
326 It has the same properties and methods as the CIM class it represents.
- 327 **3.31**  
328 **User Friendly instance Path**  
329 **UFiP**  
330 a unique path to an instance formed by concatenating the UFiTs of each instance from the root instance  
331 to the terminating instance  
332 The intervening forward slash (/) or backward slash (\) between each UFiT represents an addressing  
333 association.
- 334 **3.32**  
335 **User Friendly instance Suffix**  
336 **UFiS**  
337 a non-zero positive integer suffix added to a target instance [UFcT](#)

338 **3.33**339 **User Friendly instance Tag**340 **UFiT**

341 a unique instance tag within the scope of the target instance's containment class

342 A UFiT is created by adding a UFcT to a [UFIS](#) that is unique for that [UFcT](#).

343 **3.34**344 **User Friendly Tag**345 **UFT**

346 a short, user friendly tag for a CIM class name or instance

347 The two types of UFTs are [UFcTs](#) and [UFiTs](#).

## 348 4 User Friendly Class Tags

349 As described previously, a UFcT is designed to provide a short, simple, User Friendly Tag (UFT) for a  
 350 CIM class. A UFcT is a simple substitute for a CIM class name. The UFcT is the basis for UFITs  
 351 according to the containment shown in section 5. Figure 1 and Figure 2 show the logical and physical  
 352 containment relationships, respectively. Section 6 describes how to create a UFIT from a UFcT based on  
 353 the instance’s containment relationship.

354 Table 1 through Table 11 define standard UFcTs.

355 In these tables, the “CIM Class Name” column shows the CIM class name for each UFcT defined in the  
 356 “UFcT” column. CIM class names are organized according to the containment relationships defined in  
 357 section 5. The **bold** term in the “UFcT” column is the default UFcT. The non-bold terms in the “UFcT”  
 358 column list UFcTs that may be used instead of the default UFcT.

359 Implementations can choose to offer optional OEM-specific UFcTs. Implementations prefix each OEM-  
 360 specific UFcT with the “OEM” keyword followed by a vendor-unique identification string. The OEM string  
 361 portion of the prefix uniquely identifies the entity that owns and defines the OEM-specific UFcT. The string  
 362 includes a copyrighted, trademarked, or otherwise unique name that is owned by the business entity or  
 363 standards body defining the OEM-specific UFcT.

364 To enable proper UFcT tag assignments for instances, implementations choose the most derived class  
 365 offered in the SM Profile (for example, EthernetPort rather than the generic parent NetworkPort).  
 366 Implementations interpret UFcTs in a case-insensitive fashion.

367 In Table 1 through Table 11, the “Based On” column gives an explanation of how the UFcT is determined.  
 368 The value "class" in this column means that the UFcT is based on the CIM class name. If the UFcT is  
 369 based on the value of a CIM property within the class, the property name and value will be shown in the  
 370 “Based On” column. If the property is a string, the implementation interprets these strings in a case-  
 371 insensitive fashion. If the property is a value with a value map, the string provided is interpreted by the  
 372 implementation as the value map for the corresponding value of that property. Sometimes there is more  
 373 than one choice for a mapping of a UFcT to a property value. In these cases, the “Comments” column  
 374 gives some guidelines.

375 **Table 1 – UFcTs for Subclasses of CIM\_ManagedElement**

CIM Class Name	UFcT	Based On	Comments
CIM_AuthorizedPrivilege	<b>authorizedpriv</b>	class	
CIM_ConfigurationCapacity	<b>configcapacity</b>	class	
CIM_Location	<b>location</b>	class	
CIM_LogRecord	<b>record</b>	class	
CIM_Namespace	<b>namespace</b>	class	
CIM_PhysicalLocation	<b>plocation</b>	class	
CIM_Product	<b>product</b>	class	
CIM_RegisteredProfile	<b>profile</b>	class	
CIM_RegisteredSubProfile	<b>subprofile</b>	class	

376

**Table 2 – UFcTs for Subclasses of CIM\_Capabilities**

<b>CIM Class Name</b>	<b>UFcT</b>	<b>Based On</b>	<b>Comments</b>
CIM_MAPCapabilities	<b>mapcap</b>	class	
CIM_DeviceSharingCapabilities	<b>sharingcap</b>	class	
CIM_PowerManagementCapabilities	<b>pwrmgtcap</b>	class	
CIM_SoftwareInstallationServiceCapabilities	<b>swinstallsvccap</b>	class	
CIM_StorageCapabilities	<b>storagecap</b>	class	
CIM_StorageConfigurationCapabilities	<b>storagecfgcap</b>	class	
CIM_DHCPCapabilities	<b>dhcpcap</b>	class	

377

**Table 3 – UFcTs for Subclasses of CIM\_SettingData**

<b>CIM Class Name</b>	<b>UFcT</b>	<b>Based On</b>	<b>Comments</b>
CIM_BootConfigSetting	<b>bootcfgsetting</b>	class	
CIM_BootSourceSetting	<b>bootsrcsetting</b>	class	
CIM_BootSettingData	<b>bootsetting</b>	class	
CIM_DisplaySetting	<b>displaysetting</b>	class	
CIM_MAPSetting	<b>mapsetting</b>	class	
CIM_NicSetting	<b>nicsetting</b>	class	
CIM_StorageSetting	<b>storagesetting</b>	class	
CIM_IPAssignmentSettingData	<b>ipsettings</b>	class	
CIM_StaticIPAssignmentSettingData	<b>staticipsettings</b>	class	
CIM_DHCPSettingData	<b>dhcpsettings</b>	class	
CIM_DNSSettingData	<b>dnssettings</b>	class	
CIM_DNSGeneralSettingData	<b>dnsgeneralsettings</b>	class	

378

**Table 4 – UFcTs for Subclasses of CIM\_Collection**

CIM Class Name	UFcT	Based On	Comments
CIM_Group	<b>group</b>	class	
CIM_RedundancySet	<b>redundancysset</b>	class	
CIM_SoftwareRepository	<b>swrepo</b>	class	
CIM_ConcreteCollection	<b>concretecollection</b>	class	
	hdwr	ElementName= "Hardware"	
	capabilities	ElementName = "Capabilities"	
	capacities	ElementName = "Capacities"	
	consoles	ElementName = "Consoles"	
	logs	ElementName = "Logs"	
	profiles	ElementName = "Profiles"	
	privileges	ElementName = "Privileges"	
	products	ElementName = "Products"	
	settings	ElementName = "Settings"	
sensors	ElementName = "Sensors"		

379

**Table 5 – UFcTs for Subclasses of CIM\_LogicalElement**

CIM Class Name	UFcT	Based On	Comments
CIM_SoftwareIdentity	<b>swid</b>	class	
CIM_StoragePool	<b>storagepool</b>	class	

380

**Table 6 – UFcTs for Subclasses of CIM\_EnabledLogicalElement**

CIM Class Name	UFcT	Based On	Comments
CIM_Account	<b>account</b>	class	
CIM_ConcreteJob	<b>job</b>	class	
CIM_JobQueue	<b>jobq</b>	class	
CIM_RecordLog	<b>log</b>	class	
CIM_OperatingSystem	<b>os</b>	class	

381

Table 7 – UFcTs for Subclasses of CIM\_System

CIM Class Name	UFcT	Based On	Comments
CIM_AdminDomain	<b>admin</b>	class	
CIM_ComputerSystem	<b>system</b>	class	
	modular	OtherDedicatedDescriptions= "Modular"	Dedicated= "Other"
	storage	Dedicated= "Storage"	
	router	Dedicated= "Router"	
	switch	Dedicated= "Switch"	
	hub	Dedicated= "Hub"	
	firewall	Dedicated= "Firewall"	
	printserver	Dedicated= "Print"	
	accessserver	Dedicated= "Access Server"	
	ioserver	Dedicated= "I/O"	
	webcache	Dedicated= "Web Caching"	
	management	Dedicated= "Management"	
	blockserver	Dedicated= "Block Server"	
	fileserver	Dedicated= "File Server"	
	mobile	Dedicated= "Mobile User Device"	
	repeater	Dedicated= "Repeater"	
	bridge	Dedicated= "Bridge/Extender"	An implementation may choose either bridge or extender.
	extender	Dedicated= "Bridge/Extender"	An implementation may choose either bridge or extender.
	gateway	Dedicated= "Gateway"	
	storagevlizer	Dedicated= "Storage Virtualizer"	
	medialib	Dedicated= "Media Library"	
	nashead	Dedicated= "NAS Head"	
	nas	Dedicated= "Self-contained NAS"	
ups	Dedicated= "UPS"		
ipphone	Dedicated= "IP Phone"		
map	Dedicated= "Manageability Access Point"		
sp	Dedicated= "Management Controller"		
chassismgr	Dedicated= "Chassis Manager"		



**Table 8 – UFcTs for Subclasses of CIM\_LogicalDevice**

<b>CIM Class Name</b>	<b>UFcT</b>	<b>Based On</b>	<b>Comments</b>
CIM_AlarmDevice	<b>alarm</b>	class	
CIM_Battery	<b>battery</b>	class	
CIM_CDROMDrive	<b>cd</b>	class	
CIM_CoolingDevice	<b>cooling</b>	class	
CIM_DAPort	<b>daport</b>	class	
CIM_DiskDrive	<b>diskdrive</b>	class	
CIM_DisketteDrive	<b>floppy</b>	class	
CIM_DiskPartition	<b>diskpartition</b>	class	
CIM_Display	<b>display</b>	class	
CIM_DVDDrive	<b>dvd</b>	class	
CIM_Fan	<b>fan</b>	class	
CIM_FCPort	<b>fcport</b>	class	
CIM_HeatPipe	<b>heatpipe</b>	class	
CIM_IBPort	<b>ibport</b>	class	
CIM_Keyboard	<b>keyboard</b>	class	
CIM_LogicalDisk	<b>disk</b>	class	
CIM_LogicalModule	<b>logicalmodule</b>	class	
	devicetray	LogicalModuleType = "Device Tray"	
	linecard	LogicalModuleType = "Line Card"	
	blademodule	LogicalModuleType = "Blade"	
CIM_LogicalPort	<b>logicalport</b>	class	
CIM_MediaAccessDevice	<b>mediaaccess</b>	class	
CIM_Memory	<b>memory</b>	class	
CIM_Modem	<b>modem</b>	class	
CIM_NetworkPort	<b>netport</b>	class	
CIM_WirelessPort	<b>wifiport</b>	class	
CIM_EthernetPort	<b>enetport</b>	class	
CIM_PCIDevice	<b>pcidev</b>	class	
CIM_PCIBridge	<b>pcibridge</b>	class	
CIM_PointingDevice	<b>pointer</b>	class	
	mouse	PointingType= "Mouse"	
	trackball	PointingType= "Track Ball"	
	touchpad	PointingType= "Touch Pad"	
	touchscreen	PointingType= "Touch Screen"	

CIM Class Name	UFcT	Based On	Comments
CIM_PortController	<b>portctrl</b>	class	
	nic	ControllerType= "Ethernet"	
	hca	ControllerType= "IB"	
	tca	ControllerType= "IB"	
	hba	ControllerType= "FC"	
CIM_PowerSupply	<b>pwrsupply</b>	class	
CIM_Printer	<b>printer</b>	class	
CIM_Processor	<b>cpu</b>	class	
CIM_Refrigeration	<b>refrigeration</b>	class	
CIM_SCSIProtocolController	<b>scsiprotctrl</b>	class	
CIM_Sensor	<b>sensor</b>	class	
	currentsensor	SensorType= "Current"	
	tachsensor	SensorType= "Tachometer"	
	tempsensor	SensorType= "Temperature"	
	voltsensor	SensorType= "Voltage"	
	countersensor	SensorType= "Counter"	
	switchsensor	SensorType= "Switch"	
	locksensor	SensorType= "Lock"	
	humiditiesensor	SensorType= "Humidity"	
	airsensor	SensorType= "Air Flow"	
	presencesensor	SensorType= "Presence"	
	smokesensor	SensorType= "Smoke Detection"	
CIM_NumericSensor	<b>numsensor</b>	class	
	ncurrentsensor	SensorType= "Current"	
	ntachsensor	SensorType= "Tachometer"	
	ntempsensor	SensorType= "Temperature"	
	nvoltsensor	SensorType= "Voltage"	
	ncountersensor	SensorType= "Counter"	
	nswitchsensor	SensorType= "Switch"	
	nlocksensor	SensorType= "Lock"	
	nhumiditiesensor	SensorType= "Humidity"	
	nairsensor	SensorType= "Air Flow"	
	npresencesensor	SensorType= "Presence"	
	nsmokesensor	SensorType= "Smoke Detection"	
CIM_SPIPort	<b>spiport</b>	class	
CIM_StorageVolume	<b>storagevol</b>	class	
CIM_StorageExtent	<b>storageext</b>	class	
CIM_SerialPort	<b>serialport</b>	class	
CIM_TapeDrive	<b>tapedrive</b>	class	

CIM Class Name	UFcT	Based On	Comments
CIM_USBPort	usbport	class	
CIM_WatchDog	watchdog	class	
CIM_PortController	portctrl	class	

383

**Table 9 – UFcTs for Subclasses of CIM\_Service**

CIM Class Name	UFcT	Based On	Comments
CIM_BootService	bootsvc	class	
CIM_CLPService	clpsvc	class	
CIM_IPConfigurationService	ipcfgsvc	class	
CIM_PowerManagementService	pwrmgtsvc	class	
CIM_SharedDeviceManagementService	shareddevicesvc	class	
CIM_SoftwareInstallationService	swinstallsvc	class	
CIM_SSHService	sshsvc	class	
CIM_StorageConfigurationService	storagecfgsvc	class	
CIM_TelnetService	telnetsvc	class	
CIM_TextRedirectionService	textredirectsvc	class	
CIM_TimeService	timesvc	class	

384

**Table 10 – UFcTs for Subclasses of CIM\_ServiceAccessPoint**

CIM Class Name	UFcT	Based On	Comments
CIM_ProtocolEndpoint	protoendpt	class	
CIM_IPProtocolEndpoint	ipendpt	class	
CIM_DHCPProtocolEndpoint	dhcpendpt	class	
CIM_DNSProtocolEndpoint	dnsendpt	class	
CIM_RemoteServiceAccessPoint	remotesap	class	
	dnsserver	AccessContext="DNS Server"	
	dhcpserver	AccessContext="DHCP Server"	
	gateway	AccessContext="Default Gateway"	
CIM_LANEndpoint	lanendpt	class	
CIM_RemotePort	remoteport	class	
CIM_SCSIProtocolEndPoint	scsiendpt	class	
CIM_ServiceAccessURI	serviceuri	class	
CIM_TextRedirectionServiceAccessPoint	textredirectsap	class	

385

Table 11 – UFcTs for Subclasses of CIM\_PhysicalElement

CIM Class Name	UFcT	Based On	Comments
CIM_PhysicalPackage	<b>pkg</b>	class	
	bladepkg	PackageType= "Blade"	
	bladexpkg	PackageType= "Blade Expansion"	
	diskpkg	PackageType= "Storage Media Package"	
	fanpkg	PackageType= "Fan"	
	pwrpkg	PackageType= "Power Supply"	
	rackpkg	PackageType= "Rack"	
	chassispkg	PackageType= "Chassis/Frame"	If the package is a chassis rather than a frame, choose chassispkg.
	framepkg	PackageType= "Chassis/Frame"	If the package is a frame rather than a chassis, choose framepkg.
	backplanepkg	PackageType= "Crossconnect/Backplane"	
	sensorpkg	PackageType= "Sensor"	
	modulepkg	PackageType= "Module/Card"	If the package is a module rather than a card, choose modulepkg.
	cardpkg	PackageType= "Module/Card"	If the package is a card rather than a module, choose cardpkg.
	batterypkg	PackageType= "Battery"	
	cpupkg	PackageType= "Processor"	
	memorypkg	PackageType= "Memory"	
storagepkg	PackageType= "Storage Media Package"		
pwrsrcpkg	PackageType= "Power Source/Generator"		
CIM_PhysicalFrame	<b>frame</b>	class	
CIM_Rack	<b>rack</b>	class	
CIM_Chassis	<b>chassis</b>	class	
	laptop	ChassisPackageType= "LapTop"	
	desktop	ChassisPackageType= "Desktop"	
	tower	ChassisPackageType= "Tower"	
	storagechas	ChassisPackageType= "Storage Chassis"	
	notebook	ChassisPackageType= "Notebook"	
	mainchassis	ChassisPackageType= "Main"	

CIM Class Name	UFcT	Based On	Comments
		System Chassis"	
	expansion	ChassisPackageType= "Bus Expansion Chassis"	
	peripheralchassis	ChassisPackageType= "Peripheral Chassis"	
	subchassis	ChassisPackageType= "SubChassis"	
CIM_Card	<b>card</b>	class	
CIM_SystemBusCard	<b>buscard</b>	class	
	pcicard	BusType= "PCI"	
	eisacard	BusType= "EISA"	
	vesacard	BusType= "VESA"	
	pcmcia	BusType= "PCMCIA"	The UFcT does not discriminate among PCMCIA bus card types.
		BusType= "PCMCIA Type I"	
		BusType= "PCMCIA Type II"	
		BusType= "PCMCIA Type III"	
	accesscard	BusType= "Access.bus"	
	nubuscard	BusType= "NuBus"	
	agpcard	BusType= "AGP"	
	vmecard	BusType= "VME Bus"	
	pccard	BusType= "PC-98", "PC-98-Hireso", "PC-H98", "PC-98Note", "PC-98Full"	
	pcixcard	BusType= "PCI-X"	
	pciecard	BusType= "PCI-E"	
	sbuscard	BusType= "Sbus IEEE 1396-1993 32 bit", "Sbus IEEE 1396-1993 64 bit"	
	isacard	BusType= "ISA"	
	mcacard	BusType= "MCA"	
	giocard	BusType= "GIO"	
	xiocard	BusType= "XIO"	
	hiocard	BusType= "HIO"	
	pmccard	BusType= "PMC"	
	ibcard	BusType= "Infiniband"	
CIM_PhysicalComponent	<b>component</b>	class	
CIM_Chip	<b>chip</b>	class	
	propchip	FormFactor= "Proprietary Chip"	
	sip	FormFactor= "SIP"	
	dip	FormFactor= "DIP"	
	zip	FormFactor= "ZIP"	
	soj	FormFactor= "SOJ"	
	simm	FormFactor= "SIMM"	
	dimmm	FormFactor= "DIMM"	

CIM Class Name	UFcT	Based On	Comments
	tsop	FormFactor= "TSOP"	
	pga	FormFactor= "PGA"	
	rimm	FormFactor= "RIMM"	
	sodimm	FormFactor= "SODIMM"	
	srimm	FormFactor= "SRIMM"	
	smd	FormFactor= "SMD"	
	ssmp	FormFactor= "SSMP"	
	qfp	FormFactor= "QFP"	
	tqfp	FormFactor= "TQFP"	
	soic	FormFactor= "SOIC"	
	lc	FormFactor= "LCC"	
	plcc	FormFactor= "PLCC"	
	bga	FormFactor= "BGA"	
	fpbga	FormFactor= "FPBGA"	
	lga	FormFactor= "LGA"	
CIM_PhysicalMemory	<b>pmem</b>	class	
	ram	MemoryType= "RAM"	
	dram	MemoryType= "DRAM"	
	synchdram	MemoryType= "Synchronous DRAM"	
	cache	MemoryType= "Cache DRAM"	
	edo	MemoryType= "EDO"	
	edram	MemoryType= "EDRAM"	
	vram	MemoryType= "VRAM"	
	sram	MemoryType= "SRAM"	
	flash	MemoryType= "Flash"	
	EEPROM	MemoryType= "EEPROM"	
	EPROM	MemoryType= "EPROM"	
	cdram	MemoryType= "CDRAM"	
	sdram	MemoryType= "SDRAM"	
	sgram	MemoryType= "SGRAM"	
	rdram	MemoryType= "RDRAM"	
	ddr	MemoryType= "DDR"	
bram	MemoryType= "BRAM"		
CIM_PhysicalConnector	<b>connector</b>	class	
CIM_Slot	<b>slot</b>	class	

## 386 5 SM Containment Hierarchy

387 This section describes the SM CIM-based containment hierarchy that forms the foundation of the SM ME  
388 addressing standard.

### 389 5.1 General Concepts

390 The physical containment hierarchy defines how PhysicalElement classes are addressed. The logical  
391 containment hierarchy defines how LogicalElement classes are addressed. The SM containment  
392 hierarchy is based on [CIM Schema 2.10](#) and will be revised to accommodate later CIM Schema versions  
393 as required.

394 CIM divides its schema into two realms: logical objects and physical objects. All logical objects derive  
395 from the LogicalElement class. All physical objects derive from the PhysicalElement class.

396 In CIM, all aspects of an object that are not related to its three-dimensional, concrete real world existence  
397 are modeled as LogicalElement instances. This includes devices. A CIM\_PhysicalElement (PE) is an  
398 object that occupies space and can be touched. A LogicalElement, on the other hand, does not occupy  
399 space and cannot be touched. This label might seem counterintuitive for LogicalElement subclasses such  
400 as ComputerSystem and LogicalDevice. The ComputerSystem is the LogicalElement that comprises all  
401 the functionality of a system — not just the hardware, but the software, firmware, processes, file systems,  
402 Internet addresses, logs, and so on. In a single-chassis system one can imagine that one is touching the  
403 entire computer system, but in reality one is touching only the form factor that the computer is enclosed  
404 within. Similarly, one may argue that a device is a tangible thing. One can touch a disk drive. As in the  
405 previous example, the disk drive is the *physical* form factor (a PhysicalPackage) that encloses the *logical*  
406 aspects of a disk, such as storage volumes, disk partitions, and memory blocks. In CIM, only the form  
407 factor is a PhysicalPackage, and all the rest, including the logical entity that describes the kind of disk  
408 drive (for example, floppy, CD, DVD), are LogicalDevice instances.

409 The SM ME addressing scheme is based on the hierarchical pattern of associations between CIM  
410 objects. A UFiP is created by identifying the specific object (class or instance) tags (UFITs) to a leaf object  
411 starting from the root object (AdminDomain). SM ME addressing uses a forward slash (/) or a backward  
412 slash (\) between object terms to signify that an addressing association exists between the terms. The SM  
413 containment hierarchies define the addressing associations. The addressing associations are listed in  
414 5.5. The SM ME addressing containment relationships are listed in Table 12 and Table 13. These  
415 relationships are stated more formally in 6.3.

416 Another concept worth noting is the difference between a class and an instance of a class. A class is the  
417 template that one uses to create an instance. The class defines the nature of the properties and methods:  
418 data type, size, range, signature (input, output), and so on. An instance is a unique instantiation of the  
419 class template with specific information for each variable term. So, when one uses a class term (UFcT),  
420 such as cpu or disk, one is addressing a particular class. When one uses an instance term (UFiT), one is  
421 addressing a particular object. For example, cpu1 or disk3 contain specific information for one particular  
422 CPU or disk.

### 423 5.2 Logical Containment

424 The root of the SM MAP address space is the AdminDomain. The term “root” in this context denotes the  
425 highest order container in which the MAP stores information. Each class in the hierarchy is considered to  
426 be a container of its own information, including association classes that contain information regarding the  
427 classes they associate.

428 The ComputerSystem instance is the root of its address space. That is, all the information about the  
429 ManagedElement instances within a single computer system is contained within the ComputerSystem  
430 instance through associations to the contained instances.

431 The UML diagram in Figure 1 defines the SM logical containment hierarchy. At the top of the diagram is  
 432 the AdminDomain class. For each ComputerSystem instance managed by a MAP, a  
 433 <SystemComponent> association exists between the AdminDomain instance and the ComputerSystem  
 434 instance. From here the client can traverse intervening addressing associations to address SM MEs. It is  
 435 the job of the parser to validate that each UFIP is valid according to the grammar explained in 6.2.

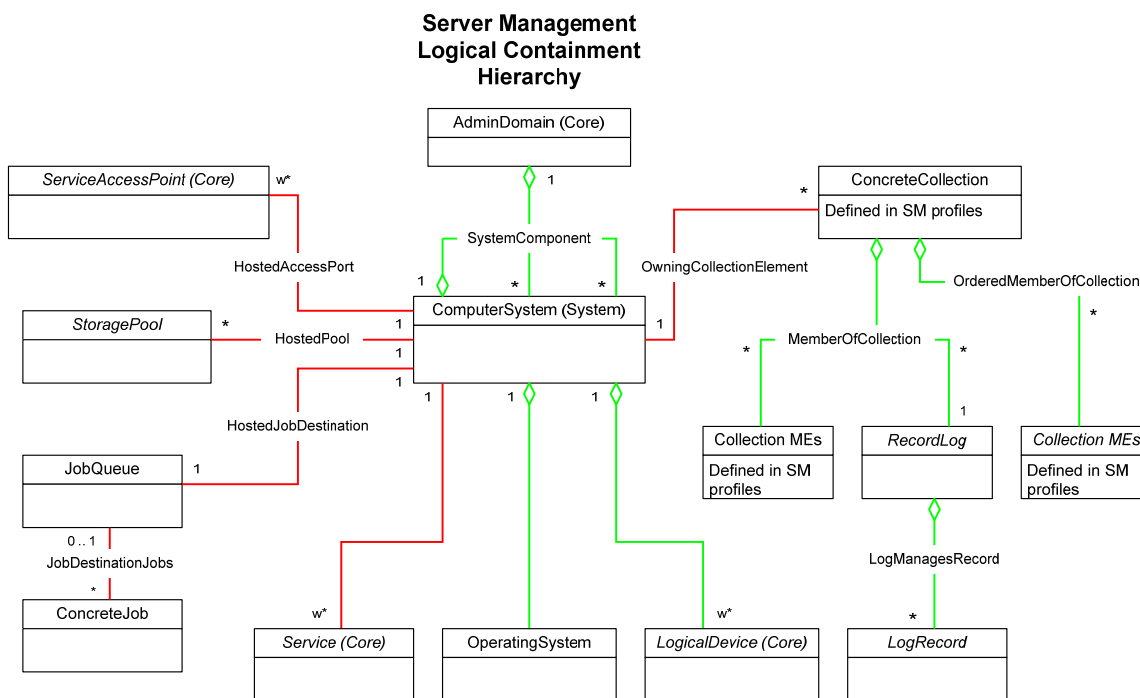
436 In Figure 1, we see that SM logical containment hierarchy provides the following associations to address  
 437 logical entities within an instance of the ComputerSystem class and to define their containment  
 438 relationship. The ← notation denotes the direction of containment. The container is on the left of the  
 439 arrow head and is read as “contains”. The addressing association is in <italics>.

440 NOTE: This is not an exhaustive list, and the information in Figure 1 and Figure 2 is not exhaustive.

- 441 ComputerSystem ← <HostedService> ← Service
- 442 ComputerSystem ← <HostedAccessPoint> ← ServiceAccessPoint
- 443 ComputerSystem ← <OwningCollectionElement> ← ConcreteCollection
- 444 ComputerSystem ← <HostedPool> ← StoragePool
- 445 ComputerSystem ← <SystemComponent> ← ComputerSystem
- 446 ComputerSystem ← <SystemDevice> ← LogicalDevice

447 The complete lists of allowable managed elements are in Table 1 through Table 11 and section 6.

448 Note that the cardinality in the CIM Schema may be different than what is represented in Figure 1 and  
 449 Figure 2. In order to enforce containment, this specification restricts the cardinality of the aggregations to  
 450 what is shown in Figure 1 and Figure 2.



451

452

Figure 1 – SM Logical Containment UML Diagram



453 The relationships described in the previous paragraph can be expressed in the following hierarchical  
454 notation:

455 AdminDomain/ComputerSystem

456 AdminDomain/ComputerSystem/Service

457 AdminDomain/ComputerSystem/ServiceAccessPoint

458 AdminDomain/ComputerSystem/ConcreteCollection

459 AdminDomain/ComputerSystem/ConcreteCollection/ME

460 AdminDomain/ComputerSystem/ConcreteCollection/RecordLog

461 AdminDomain/ComputerSystem/ConcreteCollection/RecordLog/LogRecord

462 AdminDomain/ComputerSystem/LogicalDevice

463 AdminDomain/ComputerSystem/ComputerSystem/LogicalDevice

464 In the preceding list, the forward slash (/) and backward slash (\) notation indicates the intervening  
465 addressing association. Section 6.1 describes how a UFiT is created from a UFcT. The resultant UFiTs  
466 can then be substituted for the CIM class names in the preceding list to form distinct SM ME instance  
467 addresses. Multiple addressing associations may associate an ME to its containers. The implementation  
468 selects exactly one of these addressing associations to use for addressing all instances of the ME.

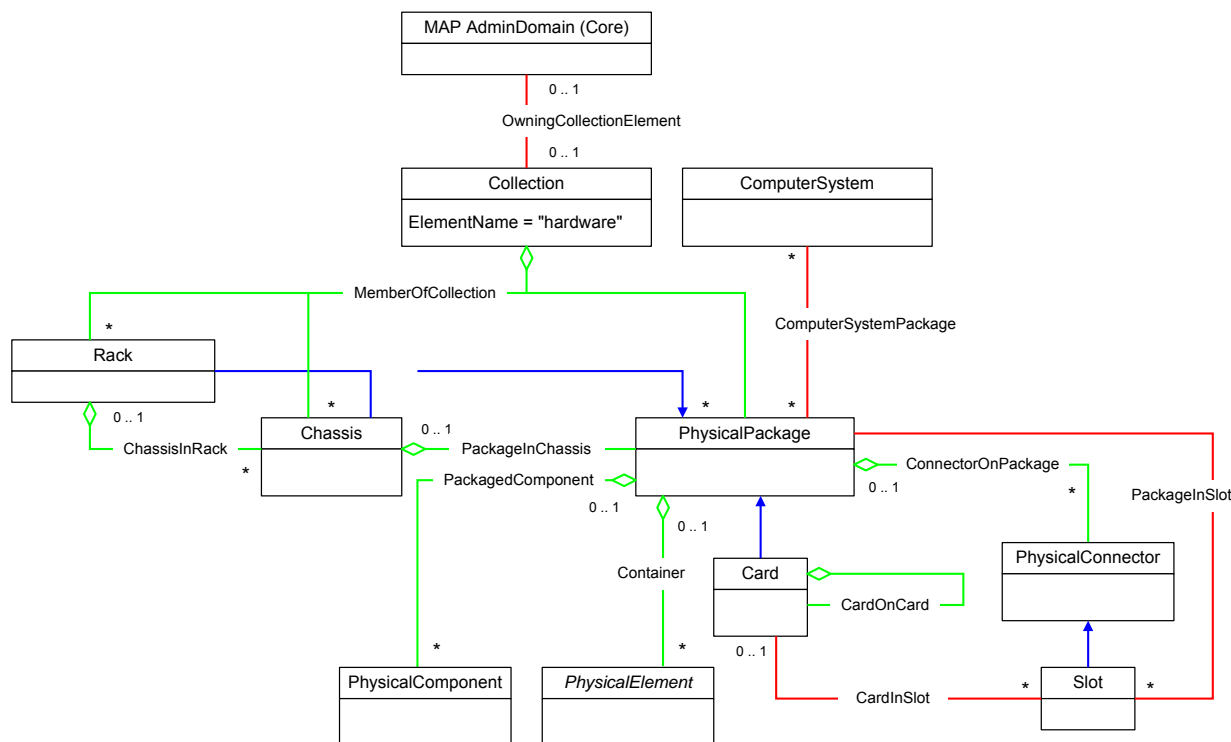
469 In Figure 1, CIM defines that a single AdminDomain instance may contain 0 to  $n$  ComputerSystem  
470 instances and 0 to  $n$  Collection instances. A ComputerSystem instance contained in the AdminDomain  
471 instance may contain 0 to  $n$  Service instances; 0 to  $n$  ServiceAccessPoint instances; 0 to  $n$  Collection  
472 instances; 0 to  $n$  LogicalDevice instances; 0 to  $n$  StoragePool instances; and 0 to  $n$  contained  
473 ComputerSystem instances that also may contain 0 to  $n$  LogicalDevice instances, Service instances, and  
474 so on. This observation forms the basis of the addressing rules defined in section 6.

475 The ConcreteCollection class represents all the subclasses of that class that appear in the SM profiles,  
476 including SoftwareRepository, RedundancySet, Group, and ConcreteCollection. The term “Collection  
477 MEs” associated to the Collection through MemberOfCollection or OrderedMemberOfCollection  
478 represents the MEs that are contained in the Collection (or a subclass) instance.

479 Similarly, ComputerSystem, ServiceAccessPoint, LogicalDevice, and Service represent all the subclasses  
480 of these classes that appear in the SM profiles.

### 481 **5.3 Physical Containment**

482 The example in Figure 2 shows the SM physical containment hierarchy and the association hierarchy that  
483 defines the SM ME addressing scheme and the containment associations for server hardware managed  
484 in the MAP’s AdminDomain class.



485

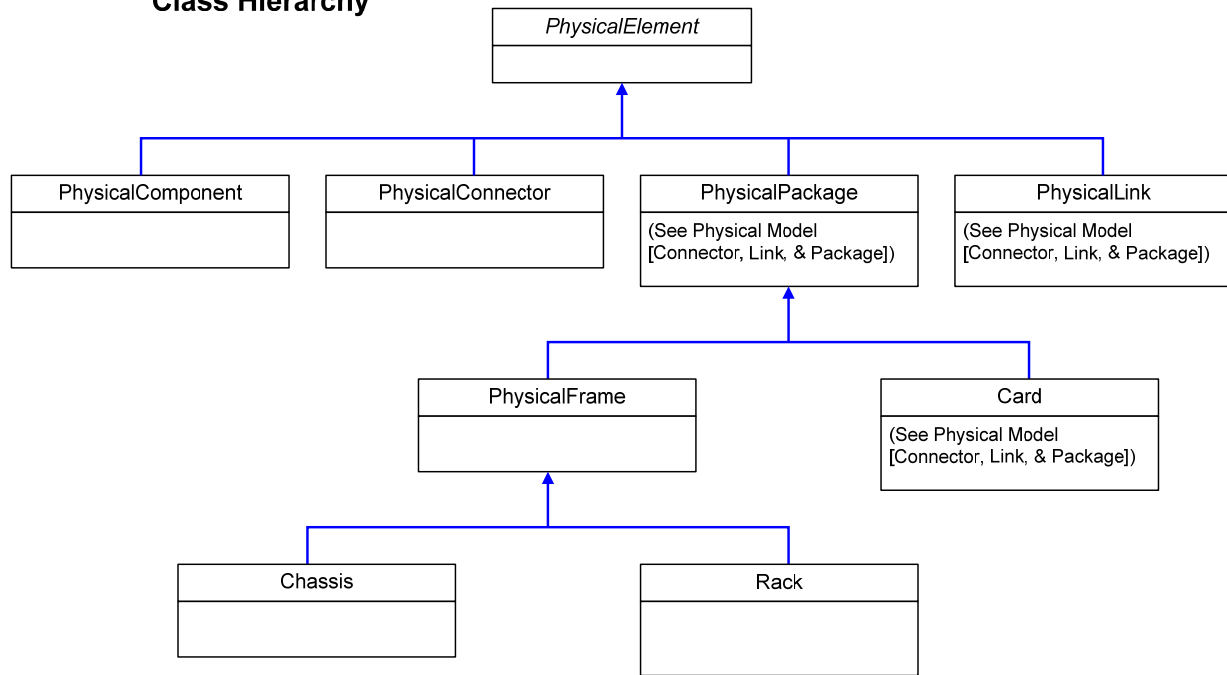
486

**Figure 2 – SM Physical Containment UML Diagram**

487 CIM defines several aggregation associations for PEs. SM addressing leverages the rules of the  
 488 associations defined in Figure 2 to uniquely address any PE. Figure 2 shows that the MAP AdminDomain  
 489 contains a single ConcreteCollection instance with the ElementName property set to “hardware”. Through  
 490 the *<MemberOfCollection>* association, this collection contains the topmost PhysicalPackage instances  
 491 of a ComputerSystem instance that the MAP manages. If this is a Rack object, then it is the top of the  
 492 hierarchy that contains all the PEs in that Rack. A Chassis within this Rack would be related with a  
 493 *<ChassisInRack>* association and the PhysicalPackage instance in each Chassis instance would be  
 494 related with a *<PackageInChassis>* association. Similarly, if the topmost PhysicalPackage instance of a  
 495 ComputerSystem managed by the MAP is a Chassis or PhysicalPackage, the enclosed PEs would use  
 496 the associations described in Figure 2.

497 A ComputerSystem instance may be composed of hardware components that are contained in a single  
 498 form factor (Chassis) or components that span multiple Racks. Figure 2 shows that the ComputerSystem  
 499 is associated to instances of its topmost PhysicalPackage instances through the  
 500 *<ComputerSystemPackage>* association ([DPS0217](#)). Similarly, Figure 2 shows that each PE in a  
 501 ComputerSystem instance can be associated with one or more specific LogicalDevice instances using the  
 502 *<Realizes>* association. The PhysicalElement inheritance hierarchy is shown in Figure 3. For the  
 503 complete PhysicalElement hierarchy, refer to the [CIM Schema](#).

**CIM Physical Element Class Hierarchy**



504

505

**Figure 3 – CIM PhysicalElement Class Hierarchy**

506 The PhysicalElement inheritance hierarchy defines that a PE may be an instance of PhysicalPackage  
 507 (frame, rack, and chassis), Component, Connector, or a PhysicalLink. Because a Chassis is a  
 508 PhysicalPackage it may be substituted for a PhysicalPackage in the physical containment diagram shown  
 509 in Figure 2, adding *n* levels of recursion. A PhysicalPackage may also contain any PE, adding another  
 510 level of recursion. Thus, the potential hardware containment relationship hierarchy is *n* deep, which is  
 511 complex enough to model any arbitrary computer hardware topology. The [SM](#) profile specifications  
 512 provide guidelines for vendors on how to model their hardware platforms to conform to the containment  
 513 hierarchy. Vendors decide which components in their systems will be manageable by creating the  
 514 appropriate ME instances. The rules for ME containment are discussed in sections 5.4 and 6.

515 Figure 2 helps to illustrate how a MAP addresses the hardware components of the ComputerSystem  
 516 instances that it manages. Figure 2 shows the MAP AdminDomain top-down containment relationships as  
 517 follows:

- 518 AdminDomain ← <OwningCollectionElement> ← ConcreteCollection ← <MemberOfCollection> ←
- 519 PhysicalPackage
- 520 Rack ← <ChassisInRack> ← Chassis ← <PackageInChassis> ← PhysicalPackage
- 521 ← <PackagedComponent> ← PhysicalComponent
- 522 or
- 523 ← <ConnectorOnPackage> ← PhysicalConnector
- 524 or
- 525 ← <Container> ← PhysicalElement
- 526 or
- 527 Chassis ← <PackageInChassis> ← PhysicalPackage

528                   ← <PackagedComponent> ← PhysicalComponent  
 529                   or  
 530                   ← <ConnectorOnPackage> ← PhysicalConnector  
 531                   or  
 532                   ← <Container> ← PhysicalElement  
 533                   or  
 534                   PhysicalPackage ← <PackagedComponent> ← PhysicalComponent  
 535                   or  
 536                   ← <ConnectorOnPackage> ← PhysicalConnector  
 537                   or  
 538                   ← <Container> ← PhysicalElement

539 By definition, the MAP AdminDomain contains a single ConcreteCollection instance that contains 0 to *n*  
 540 PhysicalPackage instances (Rack, Chassis or PhysicalPackage) that represent the topmost physical  
 541 containers of the ComputerSystem instances that the MAP is managing. A single Rack instance contains  
 542 0 to *n* Chassis instances. A single Chassis instance contains 0 to *n* PhysicalPackage instances. A single  
 543 PhysicalPackage instance contains 0 to *n* PhysicalComponent, PhysicalConnector, or PhysicalElement  
 544 instances.

545 The SM containment rules in section 6 guarantee that contained elements are unique. The SM physical  
 546 containment hierarchy associations prescribe how containers are ordered. The specific association is  
 547 never ambiguous because it is explicit in the instance. CIM uses the term REF to denote a data type that  
 548 is a Reference to an instance. A REF is an object pointer that is used to link two objects in an association  
 549 ([CIM Schema 2.10](#)). If a Chassis is in a Rack, it will be the Chassis REF in a <ChassisInRack>  
 550 association. If a Chassis is in a PhysicalPackage it will be the PE REF in a <Container> association. That  
 551 is, the Chassis is either immediately contained in the Rack class or immediately contained within some  
 552 PhysicalPackage subclass (for example, a Frame or another Chassis). The SM physical containment  
 553 hierarchy shows that only a PhysicalConnector instance can have a <ConnectorOnPackage> association  
 554 to a PhysicalPackage instance and only a PhysicalComponent instance can have a  
 555 <PackagedComponent> association to a PhysicalPackage instance. Any PE may be the PE REF in a  
 556 <Container> association to a PhysicalPackage instance. However, the PE in the <Container> association  
 557 will not be a PartComponent REF in another containment association.

558 The MAP AdminDomain addressing associations and containment relationships are expressed in the  
 559 following notation, where the forward slash (/) and backward slash (\) notation indicates the intervening  
 560 association defined in Figure 2:

561       AdminDomain/ConcreteCollection/Rack/Chassis/PhysicalPackage  
 562       AdminDomain/ConcreteCollection/Rack/Chassis/PhysicalPackage/PhysicalConnector  
 563       AdminDomain/ConcreteCollection/Rack/Chassis/PhysicalPackage/PhysicalComponent  
 564       AdminDomain/ConcreteCollection/Rack/Chassis/PhysicalPackage/PhysicalElement  
 565       AdminDomain/ConcreteCollection/Chassis/PhysicalPackage  
 566       AdminDomain/ConcreteCollection/Chassis/PhysicalPackage/PhysicalConnector  
 567       AdminDomain/ConcreteCollection/Chassis/PhysicalPackage/PhysicalComponent  
 568       AdminDomain/ConcreteCollection/Chassis/PhysicalPackage/PhysicalElement  
 569       AdminDomain/ConcreteCollection/PhysicalPackage

570 AdminDomain/ConcreteCollection/PhysicalPackage/PhysicalConnector

571 AdminDomain/ConcreteCollection/PhysicalPackage/PhysicalComponent

572 AdminDomain/ConcreteCollection/PhysicalPackage/PhysicalElement

573 Note that the physical containment hierarchy can be arbitrarily deep based on the hardware system being  
574 represented.

575 An implementation may choose not to implement instances of every PE class; therefore, in those  
576 implementations, some terms in the address path may not appear. Required associations include one of  
577 the following, depending on the type of the topmost PhysicalPackage being managed:

578 AdminDomain/ConcreteCollection/Rack

579 AdminDomain/ConcreteCollection/Chassis

580 AdminDomain/ConcreteCollection/PhysicalPackage

581 MAP implementations define which PEs are available to be managed by creating instances of the PE  
582 according to the SM profiles. Section 6.1 describes how a UFiT is created from a UFcT. The resultant  
583 UFiTs can then be substituted for the CIM class names in the physical containment hierarchy to form  
584 distinct SM PE instance addresses.

#### 585 **5.4 SM ME Addressing Hierarchy**

586 In summary, the SM logical containment hierarchy uses CIM logical containment associations to address  
587 the Service, ServiceAccessPoint, Collection, ComputerSystem, StoragePool, and LogicalDevice  
588 instances in a ComputerSystem instance in the MAP's AdminDomain instance. The SM physical  
589 containment hierarchy defines how to address physical components in the MAP's AdminDomain. The  
590 addressing rules (section 6.2) define how to create a UFiT from a UFcT so that it is unique within the  
591 managed ComputerSystem.

592 The diagram in Figure 4 combines the components of the logical and physical containment hierarchies to  
593 define the SM ME addressing hierarchy.



604

**Table 12 – SM Hardware Containment Relationships**

Container Class	Addressing Association Class	Contained Class
CIM_AdminDomain	<i>CIM_OwningCollectionElement</i>	CIM_ConcreteCollection
CIM_ConcreteCollection	<i>CIM_MemberOfCollection</i>	CIM_PhysicalPackage
CIM_Rack	<i>CIM_ChassisInRack</i>	CIM_Chassis
CIM_Chassis	<i>CIM_PackageInChassis</i>	CIM_PhysicalPackage
CIM_PhysicalPackage	<i>CIM_Container</i>	CIM_PhysicalElement
CIM_PhysicalPackage	<i>CIM_PackagedComponent</i>	CIM_PhysicalComponent
CIM_PhysicalPackage	<i>CIM_ConnectorOnPackage</i>	CIM_PhysicalConnector
CIM_PhysicalConnector	<i>CIM_PackageInConnector</i>	CIM_PhysicalPackage
CIM_Slot	<i>CIM_PackageInSlot</i>	CIM_PhysicalPackage
CIM_Slot	<i>CIM_CardInSlot</i>	CIM_Card
CIM_Card	<i>CIM_CardOnCard</i>	CIM_Card

605 Table 13 lists the SM logical containment associations based on the logical containment hierarchy. These  
 606 associations are used in section 6 to derive unique instance tags for SM logical MEs.

607

**Table 13 – SM Logical Containment Relationships**

Container Class	Addressing Association Class	Contained class
CIM_AdminDomain	<i>CIM_SystemComponent</i>	CIM_ComputerSystem
CIM_ComputerSystem	<i>CIM_SystemComponent</i>	CIM_ComputerSystem
CIM_ComputerSystem	<i>CIM_SystemDevice</i>	CIM_LogicalDevice
CIM_ComputerSystem	<i>CIM_HostedService</i>	CIM_Service
CIM_ComputerSystem	<i>CIM_HostedAccessPoint</i>	CIM_ServiceAccessPoint
CIM_ComputerSystem	<i>CIM_OwningCollectionElement</i>	CIM_ConcreteCollection
CIM_ComputerSystem	<i>CIM_HostedPool</i>	CIM_StoragePool
CIM_ConcreteCollection	<i>CIM_MemberOfCollection</i>	CIM_ManagedElement
CIM_ConcreteCollection	<i>CIM_OrderedMemberOfCollection</i>	CIM_ManagedElement
CIM_BootConfigSetting	<i>CIM_ConcreteComponent</i>	CIM_SettingData
CIM_BootConfigSetting	<i>CIM_OrderedComponent</i>	CIM_BootSourceSetting
CIM_BootSourceSetting	<i>CIM_ConcreteComponent</i>	CIM_SettingData
CIM_RecordLog	<i>CIM_LogManagesRecord</i>	CIM_LogRecord
CIM_ComputerSystem	<i>CIM_HostedJobQueue</i>	CIM_JobQueue
CIM_JobQueue	<i>CIM_JobDestinationJobs</i>	CIM_ConcreteJob
CIM_ProtocolEndpoint	<i>CIM_BindsTo</i>	CIM_ProtocolEndpoint
CIM_ProtocolEndpoint	<i>CIM_RemoteAccessAvailableToElement</i>	CIM_RemoteServiceAccessPoint
CIM_EthernetPort	<i>CIM_PortImplementsEndpoint</i>	CIM_LANEndpoint
CIM_IPAssignmentSettingData	<i>CIM_OrderedComponent</i>	CIM_IPAssignmentSettingData

## 608 5.5 Addressing Associations

609 This specification not only defines a way of addressing instances of classes but also instances of  
610 associations. The SM ME addressing grammar defines the following syntax to address CIM association  
611 instances:

612 “UFiP=> <TargetAssoc> ”

613 “UFiP=> <TargetAssoc> =>UFiP”

614 The first form targets all the associations of type <TargetAssoc> that reference the instance specified in  
615 the preceding UFiP, and the second form targets the association instance that links the UFiP on each  
616 side of the expression.

617 The following example may help explain the use of this feature. The example shown in Figure 1 has these  
618 associations:

619       Admin Domain ← <SystemComponent> ← ComputerSystem ← <SystemDevice> ←  
620       LogicalDevice

621 Using the backward and forward slashes for the associations yields the following:

622       AdminDomain/ComputerSystem/LogicalDevice

623 If the LogicalDevice were a Processor, a possible UFiP for the previous example could be as follows:

624       admin1/system1/cpu1

625 To address all instances of the *SystemDevice* association that were associated with this instance of  
626 AdminDomain/ComputerSystem, the following syntax could be used:

627       admin1/system1=>systemdevice

628 To address the specific instance of *SystemDevice* that is associated with this instance of LogicalDevice  
629 under this AdminDomain/ComputerSystem, the following syntax could be used:

630       admin1/system1=>systemdevice=>admin1/system1/cpu1



631 **6 SM ME Addressing**

632 Section 6.1 describes how UFITs are created. Section 6.2 defines the addressing rules to which  
 633 implementations shall adhere. Section 6.3 shows the productions for the SM ME addressing grammar.  
 634 This grammar provides the basis to develop algorithms to programmatically parse a UFIP.

635 **6.1 Instance Tagging**

636 The User Friendly instance Tag (UFIT) is constructed by adding an instance suffix to the UFcT (**default** or  
 637 specific). The instance suffix is a positive non-zero integer and is unique within its container for any given  
 638 UFcT. It is recommended that suffixes within a container be numbered starting at one and increase in  
 639 value sequentially.

640 A unique UFIT within a container addresses a single ME instance. ANNEX A shows several examples of  
 641 how to create SM UFITs and UFIPs based on an instance’s UFcT and its container.

642 **6.2 Addressing Rules**

643 Table 14 defines the rules that implementations shall obey to support SM ME addressing.

644 **Table 14 – SM ME Addressing Rules**

Rule	Description
1	<p>Implementations shall support UFcTs.</p> <p>Implementations shall support the <b>default</b> UFcT for each instance of a class implemented as defined in the profile being implemented.</p> <p>Implementations shall derive the UFcT from the implemented class.</p> <p>Implementations shall use the standard UFcTs from Table 1 through Table 11 for the instantiated class.</p> <p>Implementations may use any of the specific UFcTs listed in Table 1 through Table 11 for the instantiated class.</p> <p>If an implementation uses the <b>default</b> UFcT, then the property that defines the specific UFcT may be null.</p> <p>If an implementation uses a specific UFcT based on a property, that property value shall not be null and shall agree with the property value listed for the specific UFcT in the “Based On” column of the Standard User Friendly class Tags tables (Table 1 through Table 11). If the property is a string, the implementation shall interpret the value of the property in a case-insensitive fashion. If the property is a value-mapped property, the interpretation shall interpret the string specified as the ValueMap for the corresponding value.</p> <p>If the optional, specific UFcT listed in Table 1 through Table 11 is based on a value that is an element in an array property, the implementation shall use the value of the first element in the array as the selector for the optional, specific UFcT.</p> <p>Implementations shall interpret all UFcTs and UFITs in a case-insensitive fashion.</p>
2	<p>Implementations shall support the following rules for creating instance tags from SM UFcT class root and container class defined in 6.1:</p> <p>User Friendly instance Tag, UFIT, shall be constructed by adding an instance suffix to the UFcT (<b>default</b> or specific).</p> <p>The instance suffix shall be a positive non-zero integer.</p> <p>The instance suffix shall be unique within its container for any given UFcT.</p> <p>The instance suffix is not required to be a sequential enumeration.</p> <p>The instance suffix is not required to begin at any given value for any container.</p>
3	<p>Implementations’ ME addresses shall comply with the syntax specified by the SM ME addressing grammar</p>

Rule	Description
	as specified in 6.3.
4	<p>Implementations shall define ME addresses according to the addressing associations defined in the SM containment hierarchy and SM profiles:</p> <p>Implementations shall provide exactly one UFiP from the AdminDomain to every ME.</p> <p>Implementations shall guarantee that a UFiP identifies exactly one ME instance.</p> <p>Implementations shall choose at most one addressing association to address an instance of an ME.</p> <p>Implementations shall use one of the addressing associations listed in Table 12 or Table 13 as the association used to define the relationship between the instance of the ME and its containing ME.</p>
5	Implementations shall not allow UFiTs to be modified by the client.
6	Implementations shall guarantee that the AdminDomain is the highest order node of the MAP's address space.
7	<p>Implementations may provide OEM-specific extensions to the standard UFcTs. The meaning of any "OEM" prefixed UFcT is outside the scope of this specification. Implementations that support OEM-specific extensions shall adhere to the following rules:</p> <p>Implementations shall prefix each OEM-specific UFcT with the keyword "OEM" followed by a vendor-unique identification string.</p> <p>The OEM string portion of the prefix shall uniquely identify the entity that owns and defines the OEM-specific UFcT.</p> <p>The string shall include a copyrighted, trademarked, or otherwise unique name that is owned by the business entity or standards body defining the OEM-specific UFcT.</p>
8	<p>Implementations are not required to persist the UFiT assignments over the life of a configuration. Implementations may provide a mechanism for the client to force UFiTs to be recycled. Implementations shall adhere to the following rules:</p> <p>Implementations shall define the conditions under which UFiTs are destroyed.</p> <p>Implementations shall define the algorithm, if any, used to recycle a UFiT.</p>

### 645 **6.3 Addressing Grammar**

646 This section describes the grammar productions needed to form valid SM ME addresses. The production  
647 rules and notation are defined in [RFC 2234](#). Implementations shall expose ME addresses that comply  
648 with the `MEAddress` production specified by the SM ME addressing grammar as specified in this section.  
649 Implementations shall use the UFcT production to address a CIM class. [RFC 2234](#) defines a modified  
650 version of Backus-Naur Form (BNF), called Augmented BNF (ABNF), which has proven popular among  
651 many Internet specifications. It balances compactness and simplicity with reasonable representational  
652 power. [RFC 2234](#) defines a formal set of rules for a formal unambiguous machine-parsable grammar.

653 The SM ME addressing grammar provides unambiguous productions that guarantee that a programmatic  
654 parser can be created to differentiate valid ME addresses from invalid ME addresses. The SM ME  
655 addressing rules provide directives to implementations to guarantee that a UFiT is unique within its  
656 immediate container.

657 The following is the SM ME addressing grammar:

```
658 ;;
659 ;; The SM ME Addressing grammar provides a scheme for validating that
660 ;; an address presented to a parser is valid according to the rules
661 ;; specified in the text of this document.
662 ;;
663 ;; In the grammar below, the valid Addressing Associations have been
664 ;; used to formulate valid address patterns by the grammar productions.
665 ;; The grammar assumes that the implementation has used the
```

```

666 ;; valid Addressing Associations named in tables above to form the
667 ;; relationships between the instances represented by the UFiT terms in
668 ;; the address.
669 ;;
670 ;; For example, the address /system1/logs1/log2/record1 is a valid
671 ;; address. The grammar specifies that a UFiT term using the UFcT
672 ;; "system" may be followed by a UFiT term using the UFcT "logs", a term
673 ;; using "logs" may be followed by a term using "log", and a term using
674 ;; "log" may be followed by a term using "record". It is up to the
675 ;; implementation to validate that the correct corresponding CIM
676 ;; associations were used in the CIM server to associate the terms in the
677 ;; address.
678
679 ;; Managed Element Address
680 MEAddress = UFiP                               ; objects
681 MEAddress =/ (UFiP "=>" TargetAssoc [ "=>" UFiP ]) ; associations
682
683 ;; User Friendly instance Paths:
684 ;; /admin1
685 ;; /admin1/<logical paths>
686 ;; /admin1/<physical paths>
687 ;; /system1
688 ;; /hdw1
689 UFiP = TermSeparator AdminDomainUFcT UFiS
690 UFiP =/ UFiP-Logical
691 UFiP =/ UFiP-Physical
692
693 ;; User Friendly instance Tags:
694 UFiS = %x31-39 *[%x30-39] ; User Friendly instance Suffix
695 UFiT = (UFcT UFiS)
696
697 ;; User Friendly class Tags:
698 ;; All the UFcTs in the Tables plus OEM.
699 UFcT = TableOneUFcT / TableTwoUFcT / TableThreeUFcT / TableFourUFcT
700 UFcT =/ TableFiveUFcT / TableSixUFcT / TableSevenUFcT / TableEightUFcT
701 UFcT =/ TableNineUFcT / TableTenUFcT / TableElevenUFcT
702 UFcT =/ OEMUFcT
703
704 TermSeparator = "\" / "/"
705
706 ;; User-Friendly instance Paths for Logical Containment Hierarchy
707 ;; (Example paths included prior to productions)
708 ;; /admin1/system1/<system components or collections>
709 ;; /admin1/system1/system1/<system components or collections>
710 ;; /admin1/system1/system1/system1/<system components or collections>
711 UFiP-Logical = [ TermSeparator AdminDomainUFcT UFiS ]
712               1*( TermSeparator ComputerSystemUFcT UFiS )
713               [ TermSeparator ( ComponentOfSystem / SystemCollection ) ]
714
715 ;; ComponentOfSystem is a production for immediately-contained elements
716 ;; of a ComputerSystem
717 ComponentOfSystem = Service / SAP / StoragePool / Device
718 ComponentOfSystem =/ OperatingSystem

```

```
719
720 ;; All ComputerSystem Devices except Sensors
721 ;; /admin1/system1/disk3
722 Device          = LogicalDeviceUFcT UFiS
723 Device          =/ SensorUFcT UFiS
724 Device          =/ EnetStack
725
726 ;; Operating System
727 ;; /admin1/system1/os1
728 OperatingSystem = OperatingSystemUFcT UFiS
729
730 ;; Service Access Points
731 ;; /admin1/system1/remotesapl
732 SAP             = ProtoEndptUFcT UFiS
733 SAP             =/ RemoteSAPUFcT UFiS
734 SAP             =/ RemotePortUFcT UFiS
735 SAP             =/ SCSIProtoEndptUFcT UFiS
736 SAP             =/ ServiceAccessURIUFcT UFiS
737 SAP             =/ TextRedirectSAPUFcT UFiS
738 SAP             =/ IPStack
739
740 ;; System services
741 ;; /admin1/system1/bootsvc1
742 Service         = BootSvcUFcT UFiS
743 Service         =/ CLPSvcUFcT UFiS
744 Service         =/ PowerManagementSvcUFcT UFiS
745 Service         =/ SharedDeviceMgmtSvcUFcT UFiS
746 Service         =/ SoftwareInstallationSvcUFcT UFiS
747 Service         =/ SSHSvcUFcT UFiS
748 Service         =/ StorageCfgSvcUFcT UFiS
749 Service         =/ TelnetSvcUFcT UFiS
750 Service         =/ TextRedirectSvcUFcT UFiS
751 Service         =/ TimeSvcUFcT UFiS
752 Service         =/ IpCfgSvcUFcT UFiS
753
754 ;; System storage pool
755 ;; /admin1/system1/storagepool2
756 StoragePool     = StoragePoolUFcT UFiS
757
758 ;;
759 ;; SystemCollection is a production for named collections that
760 ;; are dedicated to a ComputerSystem.
761 ;;
762 SystemCollection = JobQueue / Logs
763 SystemCollection =/ SoftwareRepo / Sensors / Consoles
764 SystemCollection =/ Group / Privileges
765 SystemCollection =/ Profiles / Capabilities / Capacities
766 SystemCollection =/ Products / Settings
767
768 ;; Dedicated collection for Capabilities
769 ;; /system1/capabilities1
770 ;; /system1/capabilities1/swinstallsvccap1
771 Capabilities     = CapabilitiesUFcT UFiS
```

```

772         [ ( TermSeparator MAPCapUFcT UFiS )
773         / ( TermSeparator SharingCapUFcT UFiS )
774         / ( TermSeparator PowerMgmtCapUFcT UFiS )
775         / ( TermSeparator SoftwareInstallationSvcCapUFcT UFiS)
776         / ( TermSeparator StorageCapUFcT UFiS )
777         / ( TermSeparator StorageCfgCapUFcT UFiS )
778         / ( TermSeparator DHCPCapUFcT UFiS ) ]
779
780 ;; Dedicated collection for Capacities
781 ;; /system1/capacities1
782 ;; /system1/capacities1/configcapacity23
783 Capacities      = CapacitiesUFcT UFiS
784                 [ ( TermSeparator ConfigCapacityUFcT UFiS ) ]
785
786 ;; Dedicated collection for system Consoles
787 ;; /system1/consoles1
788 ;; /system1/consoles1/textredirectsap1
789 Consoles        = ConsolesUFcT UFiS
790                 [ ( TermSeparator TextRedirectSAPUFcT UFiS ) ]
791
792 ;; Ethernet Stack Production
793 EnetStack = EthernetPortUFcT UFiS TermSeparator IPStack
794
795 ;; Dedicated collection for Group
796 ;; /admin1/system1/group5
797 ;; /admin1/system1/group5/account2
798 Group           = GroupUFcT UFiS
799                 [ ( TermSeparator AccountUFcT UFiS ) ]
800
801 ;; Job queue
802 ;; /system1/jobq1
803 ;; /system1/jobq1/job323
804 JobQueue        = JobQueueUFcT UFiS
805                 [ ( TermSeparator ConcreteJobUFcT UFiS ) ]
806
807 ;; Dedicated collection for Logs
808 ;; /system1/logs1
809 ;; /system1/logs1/log1
810 ;; /system1/logs1/log1/record45
811 Logs            = LogsUFcT UFiS
812                 [ ( TermSeparator RecordLogUFcT UFiS )
813                 [ ( TermSeparator LogRecordUFcT UFiS ) ] ]
814
815 ;; Dedicated collection for Privileges
816 ;; /system1/privileges1
817 ;; /system1/privileges1/authorizedpriv2
818 Privileges      = PrivilegesUFcT UFiS
819                 [ (TermSeparator PrivilegeUFcT UFiS ) ]
820
821 ;; IP Stack and ProtocolEndpoint productions
822 IPStack         = LANEndPtUFcT UFiS
823                 [ TermSeparator IPEndPtUFcT UFiS
824                 [ TermSeparator ( DhcpEndPtUFcT UFiS

```

```

825             [ TermSeparator DhcpServerUFcT UFiS ])
826         / ( DnsEndPtUFcT UFiS [TermSeparator DnsServerUFcT UFiS])
827         / (GatewayUFcT UFiS)]
828
829 ;; Dedicated collection for Profiles
830 ;; /system1/profiles1
831 ;; /system1/profiles1/profile34
832 ;; /system1/profiles1/subprofile32
833 Profiles      = ProfilesUFcT UFiS
834             [ ( TermSeparator ProfileUFcT UFiS )
835             / ( TermSeparator SubProfileUFcT UFiS )
836             ]
837
838 ;; Dedicated collection for Products
839 ;; /system1/products1
840 ;; /system1/products1/product43
841 Products      = ProductsUFcT UFiS
842             [ ( TermSeparator ProductUFcT UFiS ) ]
843
844 ;; Dedicated collection for Sensors
845 ;; /system1/sensors1
846 ;; /system1/sensors1/tempsensor4
847 Sensors       = SensorsUFcT UFiS
848             [ ( TermSeparator SensorUFcT UFiS ) ]
849
850 ;; Dedicated collection for all Settings
851 ;; /system1/settings1
852 ;; /system1/settings1/displaysetting32
853 Settings      = SettingsUFcT UFiS
854             [ (SettingData
855             / (TermSeparator IPAssignmentSettings)
856             / ( TermSeparator BootConfigSetting ) ) ]
857
858 SettingData   = ( TermSeparator DisplaySettingUFcT UFiS )
859             / ( TermSeparator MAPSettingUFcT UFiS )
860             / ( TermSeparator NICSettingUFcT UFiS )
861             / ( TermSeparator StorageSettingUFcT UFiS )
862
863 ;; Dedicated collection for Boot Configuration SettingData
864 ;; /system1/settings1/bootcfgsetting1
865 ;; /system1/settings1/bootcfgsetting1/bootsetting1
866 ;; /system1/settings1/bootcfgsetting1/nicsetting1
867 ;; /system1/settings1/bootcfgsetting1/bootsrcsetting1
868 ;; /system1/settings1/bootcfgsetting1/bootsrcsetting1/bootsetting2
869 ;; /system1/settings1/bootcfgsetting1/bootsrcsetting1/displaysetting1
870 BootConfigSetting = BootConfigSettingUFcT UFiS
871             [ ( TermSeparator BootSettingDataUFcT UFiS ) /
872             ( SettingData ) /
873             ( TermSeparator BootSourceSetting ) ]
874 BootSourceSetting = BootSourceSettingUFcT UFiS
875             [ ( TermSeparator BootSettingDataUFcT UFiS ) /
876             ( SettingData ) ]
877

```

```

878 ;; IP Assignment Settings for IP Settings
879 IPAssignmentSettings = IPSettingUFcT UFiS
880     [ ( TermSeparator StaticIPSettingUFcT UFiS )
881       / ( TermSeparator DHCPSettingUFcT )
882       / ( TermSeparator DNSGeneralSettingUFcT )
883       / ( TermSeparator DNSSettingUFcT ) ]
884
885 ;; Dedicated collection for Software
886 ;; /system1/swrepol
887 ;; /system1/swrepol/swid23
888 SoftwareRepo = SoftwareRepoUFcT UFiS
889     [ ( TermSeparator SoftwareIdUFcT UFiS ) ]
890
891 ;; User-Friendly instance Paths for Physical Containment Hierarchy
892 ;; /admin1/hdwrl
893 ;; /admin1/hdwrl/<physical package paths>
894 UFiP-Physical = [ TermSeparator AdminDomainUFcT UFiS ]
895                 TermSeparator HWConcreteCollectionUFcT UFiS
896                 [ TermSeparator PhysicalPackage ]
897
898 ;; PhysicalPackage identifies those elements which the Physical
899 ;; Containment Hierarchy allows
900 ;; to be contained within another PhysicalPackage
901 PhysicalPackage = Rack / Chassis / PhysicalFrame / Package / Card
902
903 ;; /admin1/hdwrl/rack1
904 ;; /admin1/hdwrl/rack1/chassis1/...
905 ;; /admin1/hdwrl/rack1/frame1/...
906 ;; /admin1/hdwrl/rack1/card1/...
907 ;; /admin1/hdwrl/rack1/pkg1/...
908 ;; /admin1/hdwrl/rack1/<connector or component>
909 Rack = RackUFcT UFiS
910 Rack =/ RackUFcT UFiS TermSeparator PhysicalPackage
911 Rack =/ RackUFcT UFiS TermSeparator ConnectorSlot
912 Rack =/ RackUFcT UFiS TermSeparator Component
913
914 ;; /admin1/hdwrl/chassis1
915 ;; /admin1/hdwrl/chassis1/rack1/...
916 ;; /admin1/hdwrl/chassis1/frame1/...
917 ;; /admin1/hdwrl/chassis1/card1/...
918 ;; /admin1/hdwrl/chassis1/pkg1/...
919 ;; /admin1/hdwrl/chassis1/<connector or component>
920 Chassis = ChassisUFcT UFiS
921 Chassis =/ ChassisUFcT UFiS TermSeparator PhysicalPackage
922 Chassis =/ ChassisUFcT UFiS TermSeparator ConnectorSlot
923 Chassis =/ ChassisUFcT UFiS TermSeparator Component
924
925 ;; /admin1/hdwrl/frame1
926 ;; /admin1/hdwrl/frame1/card1/...
927 ;; /admin1/hdwrl/frame1/pkg1/...
928 ;; /admin1/hdwrl/frame1/<connector or component>
929 PhysicalFrame = PhysicalFrameUFcT UFiS
930 PhysicalFrame =/ PhysicalFrameUFcT UFiS TermSeparator PhysicalPackage

```

```
931 PhysicalFrame =/ PhysicalFrameUFcT UFiS TermSeparator ConnectorSlot
932 PhysicalFrame =/ PhysicalFrameUFcT UFiS TermSeparator Component
933
934 ;; /admin1/hdwrl/pkg1
935 ;; /admin1/hdwrl/pkg1/card1/...
936 ;; /admin1/hdwrl/pkg1/pkg1/...
937 ;; /admin1/hdwrl/pkg1/<connector or component>
938 Package = PackageUFcT UFiS
939 Package =/ PackageUFcT UFiS TermSeparator PhysicalPackage
940 Package =/ PackageUFcT UFiS TermSeparator ConnectorSlot
941 Package =/ PackageUFcT UFiS TermSeparator Component
942
943 ;; /admin1/hdwrl/card1
944 ;; /admin1/hdwrl/card1/card1/...
945 ;; /admin1/hdwrl/card1/<connector or component>
946 Card = ( CardUFcT / SystemCardUFcT ) UFiS
947 Card =/ ( CardUFcT / SystemCardUFcT ) UFiS TermSeparator PhysicalPackage
948 Card =/ ( CardUFcT / SystemCardUFcT ) UFiS TermSeparator ConnectorSlot
949 Card =/ ( CardUFcT / SystemCardUFcT ) UFiS TermSeparator Component
950
951 ;; /admin1/hdwrl/.../connector1
952 ;; /admin1/hdwrl/.../slot1
953 ;; /admin1/hdwrl/.../slot1/card1
954 ;; /admin1/hdwrl/.../connector1/pkg1
955 ConnectorSlot = Connector / Slot
956 Connector = ConnectorUFcT UFiS
957 Connector =/ ConnectorUFcT UFiS TermSeparator PhysicalPackage
958 Slot = SlotUFcT UFiS
959 Slot =/ SlotUFcT UFiS TermSeparator PhysicalPackage
960
961 ;; /admin1/hdwrl/.../component1
962 ;; /admin1/hdwrl/.../chip1
963 Component = ( ComponentUFcT / ChipUFcT / PhysicalMemoryUFcT ) UFiS
964
965 ;; The following sections define the terminal productions for the
966 ;; SM ME Addressing Grammar
967
968 ;; User Friendly class Tags in the Logical Containment Hierarchy
969 ;; in the order introduced in the UFcT tables in this document.
970 ;; The ordering is based upon the UFcT class' end-most subclass.
971
972 ;; special opaque OEM specific UFcT
973 OEMUFcT = "OEM" 1*VCHAR ; VCHAR: visible characters
974
975 ;; Table 1 CIM_ManagedElement UFcTs
976 TableOneUFcT = AuthorizedPrivilegeUFcT / ConfigCapacityUFcT
977 TableOneUFcT =/ LocationUFcT / LogRecordUFcT / NamespaceUFcT
978 TableOneUFcT =/ PhysicalLocationUFcT / ProductUFcT / ProfileUFcT
979 TableOneUFcT =/ SubProfileUFcT
980
981 AuthorizedPrivilegeUFcT = "authorizedpriv"
982 ConfigCapacityUFcT = "configcapacity"
983 LocationUFcT = "location"
```



```

984 LogRecordUFcT = "record"
985 NamespaceUFcT = "namespace"
986 PhysicalLocationUFcT = "plocation"
987 ProductUFcT = "product"
988 ProfileUFcT = "profile"
989 SubProfileUFcT = "subprofile"
990
991 ;; Table 2 CIM_Capabilities UFcTs
992 TableTwoUFcT = MapCapUFcT / PowerMgmtCapUFcT / SharingCapUFcT
993 TableTwoUFcT =/ SoftwareInstSvcCapUFcT / StorageCapUFcT
994 TableTwoUFcT =/ StorageCfgCapUFcT
995 TableTwoUFcT =/ DHCPCapUFcT
996
997 MAPCapUFcT = "mapcap"
998 PowerMgmtCapUFcT = "pwrmgtcap"
999 SharingCapUFcT = "sharingcap"
1000 SoftwareInstSvcCapUFcT = "swinstallsvccap"
1001 StorageCapUFcT = "storagecap"
1002 StorageCfgCapUFcT = "storagecfgcap"
1003 DHCPCapUFcT = "dhccap"
1004
1005 ;; Table 3 CIM_SettingData UFcTs
1006 TableThreeUFcT = BootConfigSettingUFcT / BootSourceSettingUFcT
1007 TableThreeUFcT =/ BootSettingDataUFcT / DisplaySettingUFcT
1008 TableThreeUFcT =/ MAPSettingUFcT / NICSettingUFcT / StorageSettingUFcT
1009 TableThreeUFcT =/ IPSettingUFcT / StaticIPSettingUFcT / DHCPSettingUFcT
1010 TableThreeUFcT =/ DNSSettingUFcT / DNSGeneralSettingUFcT
1011
1012 BootConfigSettingUFcT = "bootcfgsetting"
1013 BootSourceSettingUFcT = "bootsrcsetting"
1014 BootSettingDataUFcT = "bootsetting"
1015 DisplaySettingUFcT = "displaysetting"
1016 MAPSettingUFcT = "mapsetting"
1017 NICSettingUFcT = "nicsetting"
1018 StorageSettingUFcT = "storagesetting"
1019 IPSettingUFcT = "ipsettings"
1020 StaticIPSettingUFcT = "staticipsettings"
1021 DHCPSettingUFcT = "dhcpsettings"
1022 DNSSettingUFcT = "dnssettings"
1023 DNSGeneralSettingUFcT = "dnsgeneralsettings"
1024
1025 ;; Table 4 CIM_Collection UFcTs
1026 TableFourUFcT = CapabilitiesUFcT / CapacitiesUFcT / ConsolesUFcT
1027 TableFourUFcT =/ ConcreteCollectionUFcT / GroupUFcT / LogsUFcT
1028 TableFourUFcT =/ HWConcreteCollectionUFcT / PrivilegesUFcT
1029 TableFourUFcT =/ ProfilesUFcT / ProductsUFcT / RedundancySetUFcT
1030 TableFourUFcT =/ SettingsUFcT / SensorsUFcT / SoftwareRepoUFcT
1031
1032 ;;
1033 ;; The UFIp grammar will enforce the UFcT(s) that can be
1034 ;; contained in each collection.
1035 ;;
1036 CapabilitiesUFcT = "capabilities"

```

```
1037 CapacitiesUFcT = "capacities"
1038 ConsolesUFcT = "consoles"
1039 ConcreteCollectionUFcT = "concretecollection"
1040 GroupUFcT = "group"
1041 LogsUFcT = "logs"
1042 HWConcreteCollectionUFcT = "hdwr"
1043 PrivilegesUFcT = "privileges"
1044 ProfilesUFcT = "profiles"
1045 ProductsUFcT = "products"
1046 RedundancySetUFcT = "redundancysset"
1047 SettingsUFcT = "settings"
1048 SensorsUFcT = "sensors"
1049 SoftwareRepoUFcT = "swrepo"
1050
1051 ;; Table 5 CIM_LogicalElement UFcTs
1052 TableFiveUFcT = SoftwareIdUFcT / StoragePoolUFcT
1053
1054 SoftwareIdUFcT = "swid"
1055 StoragePoolUFcT = "storagepool"
1056
1057 ;; Table 6 CIM_EnabledLogicalElement UFcTs
1058 TableSixUFcT = AccountUFcT / ConcreteJobUFcT / JobQueueUFcT
1059 TableSixUFcT =/ OperatingSystemUFcT / RecordLogUFcT
1060
1061 AccountUFcT = "account"
1062 ConcreteJobUFcT = "job"
1063 JobQueueUFcT = "jobq"
1064 OperatingSystemUFcT = "os"
1065 RecordLogUFcT = "log"
1066
1067 ;; Table 7 CIM_System UFcTs
1068 TableSevenUFcT = AdminDomainUFcT / ComputerSystemUFcT
1069
1070 AdminDomainUFcT = "admin"
1071 ComputerSystemUFcT = "system"
1072 ComputerSystemUFcT =/ "modular"
1073 ComputerSystemUFcT =/ "accessserver" / "blockserver"
1074 ComputerSystemUFcT =/ "bridge" / "chassismgr" / "extender"
1075 ComputerSystemUFcT =/ "fileserver" / "firewall" / "gateway" / "hub"
1076 ComputerSystemUFcT =/ "ioserver" / "iphone" / "map" / "management"
1077 ComputerSystemUFcT =/ "medialib" / "mobile" / "modular" / "nas"
1078 ComputerSystemUFcT =/ "nashead" / "printserver" / "repeater"
1079 ComputerSystemUFcT =/ "router" / "sp" / "storage" / "storagevlizer"
1080 ComputerSystemUFcT =/ "switch" / "webcache" / "ups"
1081
1082 ;; Table 8 CIM_LogicalDevice UFcTs
1083 TableEightUFcT = AlarmUFcT / BatteryUFcT / CDROMDriveUFcT
1084 TableEightUFcT =/ CoolingDeviceUFcT / DAPortUFcT / DiskDriveUFcT
1085 TableEightUFcT =/ DisketteDriveUFcT / DiskPartitionUFcT / DisplayUFcT
1086 TableEightUFcT =/ DVDDriveUFcT / EthernetPortUFcT / FanUFcT / FCPortUFcT
1087 TableEightUFcT =/ HeatPipeUFcT / IBPortUFcT / KeyboardUFcT
1088 TableEightUFcT =/ LogicalDiskUFcT / LogicalModularUFcT / LogicalPortUFcT
1089 TableEightUFcT =/ MediaAccessUFcT / MemoryUFcT / ModemUFcT
```

1090 TableEightUFcT =/ NetworkPortUFcT / PCIDeviceUFcT / PCIBridgeUFcT  
1091 TableEightUFcT =/ PointingDeviceUFcT / PortControllerUFcT  
1092 TableEightUFcT =/ PowerSupplyUFcT / PrinterUFcT / ProcessorUFcT  
1093 TableEightUFcT =/ RefrigerationUFcT / SCSIProtoControlUFcT / SensorUFcT  
1094 TableEightUFcT =/ SPIPort / StorageVolumeUFcT / StorageExtentUFcT  
1095 TableEightUFcT =/ SerialPortUFcT / TapeDriveUFcT / USBPortUFcT  
1096 TableEightUFcT =/ WatchdogUFcT / WirelessPortUFcT  
1097 TableEightUFcT =/ PortCtrlUFcT  
1098  
1099 AlarmUFcT = "alarm"  
1100 BatteryUFcT = "battery"  
1101 CDROMDriveUFcT = "cd"  
1102 CoolingDeviceUFcT = "cooling"  
1103 DAPortUFcT = "daport"  
1104 DiskDriveUFcT = "diskdrive"  
1105 DisketteDriveUFcT = "floppy"  
1106 DiskPartitionUFcT = "diskpartition"  
1107 DisplayUFcT = "display"  
1108 DVDDriveUFcT = "dvd"  
1109 EthernetPortUFcT = "enetport"  
1110 FanUFcT = "fan"  
1111 FCPortUFcT = "fcport"  
1112 HeatPipeUFcT = "heatpipe"  
1113 IBPortUFcT = "ibport"  
1114 KeyboardUFcT = "keyboard"  
1115 LogicalDiskUFcT = "disk"  
1116 LogicalModularUFcT = "logicalmodule" / "blademodule" / "linecard"  
1117 LogicalModularUFcT =/ "devicetray"  
1118 LogicalPortUFcT = "logicalport"  
1119 MediaAccessUFcT = "mediaaccess"  
1120 MemoryUFcT = "memory"  
1121 ModemUFcT = "modem"  
1122 NetworkPortUFcT = "netport"  
1123 PCIDeviceUFcT = "pcidev"  
1124 PCIBridgeUFcT = "pcibridge"  
1125 PointingDeviceUFcT = "pointer" / "mouse" / "trackball"  
1126 PointingDeviceUFcT =/ "touchpad" / "touchscreen"  
1127 PortControllerUFcT = "portctrl" / "nic" / "hca" / "tca" / "hba"  
1128 PowerSupplyUFcT = "pwrsupply"  
1129 PrinterUFcT = "printer"  
1130 ProcessorUFcT = "cpu"  
1131 RefrigerationUFcT = "refrigeration"  
1132 SCSIProtoControlUFcT = "scsiprotctrl"  
1133 SensorUFcT = "sensor" / "currentsensor" / "tachsensor"  
1134 SensorUFcT =/ "tempsensor" / "voltsensor" / "humiditysensor"  
1135 SensorUFcT =/ "countersensor" / "swsensor" / "locksensor"  
1136 SensorUFcT =/ "smokesensor" / "airsensor" / "presencesensor"  
1137 SensorUFcT =/ "numsensor" / "ncurrentsensor" / "ntachsensor"  
1138 SensorUFcT =/ "ntempsensor" / "nvoltsensor" / "nhumiditysensor"  
1139 SensorUFcT =/ "ncountersensor" / "nswsensor" / "nlocksensor"  
1140 SensorUFcT =/ "nsmokesensor" / "nairsensor" / "npresencesensor"  
1141 SPIPort = "spiport"  
1142 StorageVolumeUFcT = "storageevol"

```
1143 StorageExtentUFcT = "storageext"
1144 SerialPortUFcT = "serialport"
1145 TapeDriveUFcT = "tapedrive"
1146 USBPortUFcT = "usbport"
1147 WatchdogUFcT = "watchdog"
1148 WirelessPortUFcT = "wifiport"
1149 PortCtrlUFcT = "portctrl"
1150
1151 ;; Convenience Production for all LogicalDevice instances
1152 ;; immediately contained in a System
1153 LogicalDeviceUFcT = AlarmUFcT / BatteryUFcT / CDROMDriveUFcT
1154 LogicalDeviceUFcT =/ CoolingDeviceUFcT / DAPortUFcT / DiskDriveUFcT
1155 LogicalDeviceUFcT =/ DisketteDriveUFcT / DiskPartitionUFcT / DisplayUFcT
1156 LogicalDeviceUFcT =/ DVDDriveUFcT / EthernetPortUFcT / FanUFcT
1157 LogicalDeviceUFcT =/ FCPortUFcT / HeatPipeUFcT / IBPortUFcT
1158 LogicalDeviceUFcT =/ KeyboardUFcT / LogicalDiskUFcT /
1159 LogicalDeviceUFcT =/ LogicalModularUFcT / LogicalPortUFcT
1160 LogicalDeviceUFcT =/ MediaAccessUFcT / MemoryUFcT / ModemUFcT
1161 LogicalDeviceUFcT =/ NetworkPortUFcT / PCIDeviceUFcT / PCIBridgeUFcT
1162 LogicalDeviceUFcT =/ PointingDeviceUFcT / PortControllerUFcT
1163 LogicalDeviceUFcT =/ PowerSupplyUFcT / PrinterUFcT / ProcessorUFcT
1164 LogicalDeviceUFcT =/ RefrigerationUFcT / SCSIProtoControlUFcT
1165 LogicalDeviceUFcT =/ SPIPort / StorageVolumeUFcT / StorageExtentUFcT
1166 LogicalDeviceUFcT =/ SerialPortUFcT / TapeDriveUFcT / USBPortUFcT
1167 LogicalDeviceUFcT =/ WatchdogUFcT / WirelessPortUFcT
1168 LogicalDeviceUFcT =/ PortCtrlUFcT
1169
1170 ;; Table 9
1171 TableNineUFcT = BootSvcUFcT / CLPSvcUFcT / PowerManagementSvcUFcT
1172 TableNineUFcT =/ SharedDeviceMgmtSvcUFcT / SoftwareInstallationSvc
1173 TableNineUFcT =/ SSHSvcUFcT / StorageCfgSvcUFcT / TelnetSvcUFcT
1174 TableNineUFcT =/ TextRedirectSvcUFcT / TimeSvcUFcT
1175 TableNineUFcT =/ IpCfgSvcUFcT
1176
1177 BootSvcUFcT = "bootsvc"
1178 CLPSvcUFcT = "clpsvc"
1179 PowerManagementSvcUFcT = "pwrmgtsvc"
1180 SharedDeviceMgmtSvcUFcT = "shareddevicesvc"
1181 SoftwareInstallationSvc = "swinstallsvc"
1182 SSHSvcUFcT = "sshsvc"
1183 StorageCfgSvcUFcT = "storagecfgsvc"
1184 TelnetSvcUFcT = "telnetsvc"
1185 TextRedirectSvcUFcT = "textredirectsvc"
1186 TimeSvcUFcT = "timesvc"
1187 IPCfgSvcUFcT = "ipcfgsvc"
1188
1189 ;; Table 10
1190 TableTenUFcT = ProtoEndptUFcT / RemoteSAPUFcT / RemotePortUFcT
1191 TableTenUFcT =/ SCSIProtoEndptUFcT / ServiceAccessURIUFcT
1192 TableTenUFcT =/ TextRedirectSAPUFcT
1193 TableTenUFcT =/ IPEndPtUFcT / DNSEndPtUFcT / LANEndPtUFcT
1194 TableTenUFcT =/ DhcpEndPtUFcT / DnsServerUFcT / DhcpServerUFcT
1195 TableTenUFcT =/ GatewayUFcT
```

```

1196
1197 ProtoEndptUFcT = "protoendpt"
1198 RemoteSAPUFcT = "remotesap"
1199 RemotePortUFcT = "remoteport"
1200 SCSIProtoEndptUFcT = "scsiendpt"
1201 ServiceAccessURIUFcT = "serviceuri"
1202 TextRedirectSAPUFcT = "textredirectsap"
1203 IPEndPtUFcT = "ipendpt"
1204 DNSEndPtUFcT = "dnsendpt"
1205 LANEndPtUFcT = "lanendpt"
1206 DhcpEndPtUFcT = "dhcpendpt"
1207 DnsServerUFcT = "dnsserver"
1208 DhcpServerUFcT = "dhcpserver"
1209 GatewayUFcT = "gateway"
1210
1211 ;; User Friendly class Tags in the Physical Containment Hierarchy:
1212
1213 ;; Table 11 CIM_PhysicalElement UFcTs
1214 TableElevenUFcT = CardUFcT / ChassisUFcT / ChipUFcT / ComponentUFcT
1215 TableElevenUFcT =/ ConnectorUFcT / FrameUFcT / PackageUFcT
1216 TableElevenUFcT =/ PhysicalMemoryUFcT / RackUFcT SlotUFcT
1217 TableElevenUFcT =/ SystemCardUFcT
1218
1219 CardUFcT = "card"
1220
1221 ChassisUFcT = "chassis"
1222 ChassisUFcT =/ "laptop" / "desktop" / "tower"
1223 ChassisUFcT =/ "storagechas" / "notebook" / "mainchassis"
1224 ChassisUFcT =/ "expansion" / "peripheralchassis" / "subchassis"
1225
1226 ChipUFcT = "bga" / "chip" / "dimm" / "dip" / "fpbga" / "lc"
1227 ChipUFcT =/ "lga" / "plcc" / "pga" / "propchip" / "qfp" / "rimm"
1228 ChipUFcT =/ "sip" / "simm" / "smd" / "sodimm" / "soj" / "ssmp"
1229 ChipUFcT =/ "soic" / "srimm" / "tqfp" / "tsop" / "zip"
1230
1231 ComponentUFcT = "component"
1232
1233 ConnectorUFcT = "connector"
1234
1235 PhysicalFrameUFcT = "frame"
1236
1237 ;; generic package UFcT
1238 PackageUFcT = "pkg"
1239 PackageUFcT =/ "backplanepkg" / "batterypkg" / "bladepkg" / "bladexpkg"
1240 PackageUFcT =/ "cardpkg" / "chassispkg" / "cpupkg" / "diskpkg"
1241 PackageUFcT =/ "fanpkg" / "framepkg" / "memorypkg" / "modulepkg"
1242 PackageUFcT =/ "pwrpkg" / "pwrsrcpkg" / "rackpkg" / "sensorpkg"
1243 PackageUFcT =/ "storagepkg"
1244
1245 PhysicalMemoryUFcT = "bram" / "cache" / "cdram" / "ddr" / "dram" / "edo"
1246 PhysicalMemoryUFcT =/ "edram" / "eeprom" / "eprom" / "flash" / "pmem"
1247 PhysicalMemoryUFcT =/ "ram" / "rdram" / "sdram" / "sgram" / "sram"
1248 PhysicalMemoryUFcT =/ "synchdram" / "vram"

```

```
1249 RackUFcT = "rack"
1250 SlotUFcT = "slot"
1251 SystemCardUFcT = "accesscard" / "agpcard" / "buscard" / "eisacard"
1252 SystemCardUFcT =/ "giocard" / "hiocard" / "ibcard"
1253 SystemCardUFcT =/ "isacard" / "mcacard" / "nubuscard"
1254 SystemCardUFcT =/ "pccard" / "pcicard" / "pciecard" / "pcixcard"
1255 SystemCardUFcT =/ "pcmciacard" / "pmccard" / "sbuscard"
1256 SystemCardUFcT =/ "vesacard" / "vmecard" / "xiocard"
1257 ;;
1258 ;; Associations as the address targets:
1259 TargetAssoc = "SystemComponent" / "SystemDevice"
1260 TargetAssoc =/ "HostedService" / "HostedAccessPoint"
1261 TargetAssoc =/ "OwningCollectionElement" / "HostedPool"
1262 TargetAssoc =/ "ConcreteComponent" / "MemberOfCollection"
1263 TargetAssoc =/ "OrderedMemberOfCollection" / "LogManagesRecord"
1264 TargetAssoc =/ "HostedJobQueue" / "JobDestinationJobs"
1265 TargetAssoc =/ "MemberOfCollection"
1266 TargetAssoc =/ "ChassisInRack" / "PackageInChassis"
1267 TargetAssoc =/ "Container" / "PackagedComponent"
1268 TargetAssoc =/ "ConnectorOnPackage" / "PackageInConnector"
1269 TargetAssoc =/ "PackageInSlot" / "CardInSlot" / "CardOnCard"
1270 TargetAssoc =/ "AssociatedAlarm" / "BootDevice" / "ScopedSetting"
1271 TargetAssoc =/ "AssociatedCooling" / "ElementSoftwareIdentity"
1272 TargetAssoc =/ "ServiceAvailableToElement"
1273 TargetAssoc =/ "SAPAvailableForElement"
1274 TargetAssoc =/ "DeviceSAPImplementation"
1275 TargetAssoc =/ "DiskPartitionBasedOnVolume"
1276 TargetAssoc =/ "ElementCapabilities"
1277 TargetAssoc =/ "MediaPresent" / "SAPAvailableToElement"
1278 TargetAssoc =/ "AllocatedFromStoragePool" / "Realizes"
1279 TargetAssoc =/ "UseOfMessageLog" / "OperationLog"
1280 TargetAssoc =/ "ControlledBy" / "ComputerSystemPackage"
1281 TargetAssoc =/ "InstalledSoftwareIdentity" / "HostedDependency"
1282 TargetAssoc =/ "RunningOS" / "InstalledOS"
1283 TargetAssoc =/ "AssociatedBattery"
1284 TargetAssoc =/ "SuppliesPower" / "AssociatedSensor"
1285 TargetAssoc =/ "SCSIInitiatorTargetLogicalUnitPath" / "BindsTo"
1286 TargetAssoc =/ "ActiveConnection" / "PortImplementsEndpoint"
1287 TargetAssoc =/ "ProvidesEndpoint" / "SASSAPDependency"
1288 TargetAssoc =/ "ServiceServiceDependency" / "AuthorizedTarget"
1289 TargetAssoc =/ "AuthorizedSubject" / "AccountOnSystem"
1290 TargetAssoc =/ "ProtocolControllerForUnit"
1291 TargetAssoc =/ "AssociatedPowerManagementService"
1292 TargetAssoc =/ "SharingDependency" / "ElementCapacity"
1293 TargetAssoc =/ "ServiceAffectsElement"
1294 TargetAssoc =/ "BindsTo" / "PortImplementsEndpoint"
1295 TargetAssoc =/ "RemoteAccessAvailableToElement" / "EndpointIdentity" TargetAssoc =/
1296 "OrderedComponent" / "ProtocolControllerForPort"
1297 TargetAssoc =/ "ServiceAvailableToElement"
```

**ANNEX A**  
(informative)

**Considerations for Implementation**

1298  
1299  
1300  
1301

1302 Table A-1 shows several examples of valid SM UFiTs that are based on the instance's UFcT and are  
1303 unique within the container. The first column shows the container instance's UFcT followed by the  
1304 corresponding CIM class name in parentheses. Similarly, the second column shows the contained  
1305 instance's UFcT with its corresponding CIM class name in parentheses. The third column shows the  
1306 contained instance's UFiT followed by an example of a complete UFiP. Note that the difference between  
1307 the UFcT and the UFiT is the instance suffix (a positive non-zero integer). The instance suffix must be  
1308 unique within the scope of the container class.

1309

**Table A-1 – Examples of SM UFiTs**

Container UFcT (CIM Class Name)	Contained UFcT (CIM Class Name)	Contained Instance UFiT (Complete UFiP)
admin (CIM_AdminDomain)	hdwr (CIM_ConcreteCollection)	hdwr1 /admin1/hdwr1
hdwr (CIM_ConcreteCollection)	rack (CIM_Rack)	rack1 /admin1/hdwr1/rack1
rack (CIM_Rack)	chassis (CIM_Chassis)	Chassis4 /admin1/hdwr1/rack2/chassis4 /admin1/hdwr1/rack11/chassis4
rack (CIM_Rack)	iochassis (CIM_Chassis)	iochassis4 /admin1/hdwr1/rack2/iochassis4 /admin1/hdwr1/rack11/iochassis4
chassis (CIM_Chassis)	pkg (CIM_PhysicalPackage)	Pkg2 /admin1/hdwr1/chassi4/pkg2 /admin1/hdwr1/rack1/chassis1/pkg2
chassis (CIM_Chassis)	bladepkg (CIM_PhysicalPackage)	bladepkg2 /admin1/hdwr1/chassi4/bladepkg2 /admin1/hdwr1/rack1/chassis1/bladepkg2
pkg (CIM_PhysicalPackage)	diskpkg (CIM_PhysicalPackage)	Diskpkg1 /admin1/hdwr1/pkg1/diskpkg1 /admin1/hdwr1/rack1/pkg2/diskpkg1 /admin1/hdwr1/rack1/chassis2/pkg5/diskpkg1 /admin1/hdwr1/chassis4/pkg3/diskpkg1
bladepkg (CIM_PhysicalPackage)	pwrpkg (CIM_PhysicalPackage)	pwrpkg2 pwrpkg3 /admin1/hdwr1/bladepkg2/pwrpkg2 /admin1/hdwr1/bladepkg5/pwrpkg2 /admin1/hdwr1/rack7/bladechassis2/bladepkg8/pwrpkg3
pkg (CIM_PhysicalPackage)	chip (CIM_Chip)	chip1 chip7 /admin1/hdwr1/pkg1/chip1 /admin1/hdwr1/pkg1/chip7 /admin1/hdwr1/chassis2/pkg1/chip1
bladepkg (CIM_PhysicalPackage)	chip (CIM_Chip)	chip1 /admin1/hdwr1/bladepkg2/chip1 /admin1/hdwr1/chassis4/bladepkg4/chip1

Container UFcT (CIM Class Name)	Contained UFcT (CIM Class Name)	Contained Instance UFIT (Complete UFIP)
chassis (CIM_Chassis)	chip (CIM_Chip)	chip1 /admin1/hdwr1/chassis4/chip1
card (CIM_Card)	chip (CIM_Chip)	chip1 /admin1/hdwr1/card1/chip1 /admin1/hdwr1/chassis3/card2/chip1 /admin1/hdwr1/rack18/chassis2/bladepkg8/card2/chip1
bladepkg (CIM_PhysicalPackage)	slot (CIM_Slot)	slot1 slot8 /admin1/hdwr1/bladepkg1/slot1
card (CIM_Card)	slot (CIM_Slot)	slot1 slot4 /admin1/hdwr1/bladepkg3/card1/slot1 /admin1/hdwr1/bladepkg3/card2/slot1 /admin1/hdwr1/bladepkg3/card2/slot4
chassis (CIM_Chassis)	slot (CIM_Slot)	slot1 /admin1/hdwr1/chassis2/slot1 /admin1/hdwr1/rack11/chassis2/slot1
admin (CIM_AdminDomain)	system (CIM_ComputerSystem)	system1, system2, system7, system3, router1, map1 /admin1/system1 /admin1/ map1
system (CIM_ComputerSystem)	system (CIM_ComputerSystem)	system1, map1 /admin1/system1/system1 /admin1/system1/ map1
system (CIM_ComputerSystem)	disk (CIM_LogicalDevice)	disk1 /admin1/system2/disk1 /admin1/system;2/system1/disk1
system (CIM_ComputerSystem)	service (CIM_Service)	service1, service2, service5 /admin1/system3/service5 /admin1/system1/ map1/service5
system (CIM_ComputerSystem)	remotesap (CIM_RemoteServiceAccess Point)	remotesap1, remotesap2, remotesap3, remotesap4 /admin1/system3/ remotesap4 /admin1/system1/system1/ remotesap3
system (CIM_ComputerSystem)	group (CIM_Group)	group1 /admin1/system1/group1
system (CIM_ComputerSystem)	storagepool (CIM_StoragePool)	storagepool1 /admin1/system1/storagepool1
capabilities logs settings products (CIM_ConcreteCollection)	mapcap log swid (CIM_ManagedElement)	capabilities1, logs1, settings1, products1 /admin1/system1/capabilities1/mapcap1 /admin1/system1/ logs1/log1 /admin1/system1/settings1/mapsetting1 /admin1/system1/products1/product1



## ANNEX B (informative)

1311  
1312  
1313  
1314

### Document Conventions

#### 1315 B.1 CIM\_ Prefix

1316 When referring to CIM classes, this document uses the CIM\_ prefix only when necessary for clarity. For  
1317 example, ComputerSystem, rather than CIM\_ComputerSystem, is used unless the context demands the  
1318 prefix for clarity.

#### 1319 B.2 Notation

1320 [Augmented Backus-Naur Form \(ABNF\)](#) is used in this document to describe various aspects of the SM  
1321 addressing specification. The complete grammar can be found in 6.3.

1322 The ← notation denotes the direction of containment. The container is on the left of the arrow head and is  
1323 read as “contains”.

1324 The following fonts are used to indicate specification elements:

- 1325 • Text in `courier new font` indicates productions and literal characters used in a grammar  
1326 syntax expression.
- 1327 • Text in *<italicized>* font indicates a CIM association class.

1328

1329  
1330  
1331  
1332

## **ANNEX C** (informative)

### **Change Log**

<b>Version</b>	<b>Date</b>	<b>Author</b>	<b>Description</b>
1.0.0a	11/02/2006	A. Merkin	Preliminary Standard
1.0.0	04/23/2009		DMTF Standard Release

## Bibliography

1333

1334 DMTF DSP0217, *SMASH Implementation Requirements*, 1.0.0,  
1335 [http://www.dmtf.org/standards/published\\_documents/DSP0217\\_1.0.0.pdf](http://www.dmtf.org/standards/published_documents/DSP0217_1.0.0.pdf)

1336 DMTF DSP2001, *Systems Management Architecture for Server Hardware (SMASH) Command Line*  
1337 *Protocol (CLP) Architecture White Paper*, 1.0.0,  
1338 [http://www.dmtf.org/standards/published\\_documents/DSP2001\\_1.0.1.pdf](http://www.dmtf.org/standards/published_documents/DSP2001_1.0.1.pdf)  
1339