



1

Server Management Managed Element Addressing Specification

2

3

Version 1.0.0b

4

Status: Preliminary Standard

5

Publication Date: 11/10/2006

6

DSP0215

7

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

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

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21 claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is withdrawn or modified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

24

26
27
28
29

Version 1.0c
Publication Date: November 10, 2006
DSP0215
Status: Preliminary Standard

30 **Abstract**

31 The Systems Management Architecture for Server Hardware (SMASH) initiative is a suite of
32 specifications that standardize the manageability interfaces for server hardware. The suite of
33 specifications defines an architectural framework, interfaces in the form of protocols, a discovery
34 function, an addressing scheme, and profiles for server hardware.

35 **Acknowledgments**

36 The following persons were instrumental in the development of this specification:
37 Bob Blair, Newisys; Greg Dake, IBM; Jon Hass, Dell; Jeff Hilland, HP; Steffen Hulegaard, OSA
38 Technologies; Arvind Kumar, Intel; John Leung, Intel; Jeff Lynch, IBM; Aaron Merkin, IBM;
39 Christina Shaw (editor), HP; Enoch Suen, Dell; Michael Tehranian, Sun; Perry Vincent, Intel.

Table of Contents

41	Abstract	3
42	Acknowledgments.....	3
43	1 Introduction	6
44	1.1 Target Audience	6
45	1.2 Conventions.....	6
46	1.2.1 Notation	7
47	1.3 Related Documents	7
48	1.4 Terminology	7
49	1.5 Acronyms and Abbreviations.....	9
50	2 SM ME Addressing Overview.....	10
51	2.1 Goals	10
52	2.2 Conceptual Areas Addressed.....	10
53	3 User-Friendly Class Tags.....	11
54	4 SM Containment Hierarchy.....	20
55	4.1 Logical Containment	20
56	4.2 Physical Containment.....	24
57	4.3 SM ME Addressing Hierarchy	27
58	4.4 Addressing Associations	30
59	5 SM ME Addressing Rules	32
60	5.1 Instance Tagging	32
61	5.2 Addressing Rules.....	32
62	5.2.1 Rules for Selecting Instances by UFcT.....	33
63	5.3 SM ME Addressing Grammar.....	34
64	6 Summary	49
65	6.1 SM Profiles.....	49
66	6.2 SM ME Grammar.....	49
67	6.3 CIM Model.....	49
68	Appendix A – Considerations for Implementation.....	50
69		

70

List of Figures

71	Figure 1	SM Logical Containment UML Diagram	22
72	Figure 2	SM Physical Containment UML Diagram.....	24
73	Figure 3	High Level View SM ME Address Hierarchy	27
74			

75

List of Tables

76	Table 1	UFcTs for Subclasses of CIM_ManagedElement.....	11
77	Table 2	UFcTs for Subclasses of CIM_Capabilities.....	12
78	Table 3	UFcTs for Subclasses of CIM_SettingData	12
79	Table 4	UFcTs for Subclasses of CIM_Collection	13
80	Table 5	UFcTs for Subclasses of CIM_LogicalElement	13
81	Table 6	UFcTs for Subclasses of CIM_EnabledLogicalElement	13
82	Table 7	UFcTs for Subclasses of CIM_System	14
83	Table 8	UFcTs for Subclasses of CIM_LogicalDevice	15
84	Table 9	UFcTs for Subclasses of CIM_Service.....	16
85	Table 10	UFcTs for Subclasses of CIM_ServiceAccessPoint	17
86	Table 11	UFcTs for Subclasses of CIM_PhysicalElement	17
87	Table 12	SM Hardware Containment Relationships.....	28
88	Table 13	SM Logical Containment Relationships	29
89	Table 14	SM ME Addressing Rules	32
90	Table 15	UFiT Examples	50
91			

92 **1 Introduction**

93 This document describes the Server Management (SM) Managed Element (ME) Addressing
94 standard. SM ME Addressing provides an easy, userfriendly way to address CIM objects (classes
95 and instances). This specification may be used to define valid targets for SM CLP commands [3].
96 This specification is intended for use by other protocols.

97 SM ME Addressing is based on the CIM Containment Hierarchy described in section 4. The
98 Containment Hierarchy, which is based on relevant ME classes and associations, forms the basis of
99 the address grammar described in section 5. The Containment Hierarchy also forms the basis for SM
100 profiles. SM profiles define rules and guidelines to model specific hardware platforms like a base
101 system, modular (blade) systems, and other architectural features, such as clustering, redundancy
102 (resource sharing), and virtual machines. Other SM profiles define rules and guidelines to model
103 specific management domains such as Boot Control, Power Control, Software Update, and the like.
104 These are covered in detail in the SM profiles. This addressing specification has the grammar and
105 rules associated with the referenced version of the *SMASH Implementation Requirements*.

106 The CIM associations in the SM profiles are used to define hierarchies in the CIM model that are
107 used to create specific paths to leaf- or end-node instances, in the same manner as a directory
108 structure provides a hierarchical directory path to a leaf file or directory in a file system. The slashes
109 in the SM ME Addressing notation represent CIM associations, and User-Friendly Tags (UFTs)
110 represent certain MEs. There are two types of UFTs: User-Friendly class Tags (UFcT) and User-
111 Friendly instance Tags (UFiT). This specification defines a UFcT for each class used in the SM
112 profiles. UFcTs are introduced in section 2 and further defined in section 3. A UFcT provides a
113 short, user-friendly synonym for a specific CIM class. This specification describes the rules for
114 building UFiTs based on the instance's UFcT, the containment rules described in section 4, and the
115 addressing rules described in section 5.2.

116 A fully qualified UFiT path (UFiP) addresses a single CLP command target and may be mapped to a
117 specific CIM Object Reference, thus providing support for communication between the
118 Manageability Access Point (MAP) and CIM Servers, Clients, and Providers. These features
119 combine to enable an embedded lightweight CIM Server in the Manageability Access Point.

120 **1.1 Target Audience**

121 This specification is intended to guide developers of implementations of SM ME Addressing and
122 may be used as a reference by system administrators and other users of the SM ME Addressing
123 implementation.

124 **1.2 Conventions**

125 The key phrases and words **MUST**, **MUST NOT**, **REQUIRED**, **SHALL**, **SHALL NOT**, **SHOULD**,
126 **SHOULD NOT**, **RECOMMENDED**, **MAY**, and **OPTIONAL** in this document are to be interpreted
127 as in RFC 2119.

128 When discussing CIM classes, this document uses the CIM_ prefix only when necessary for clarity.
129 For example, the text refers to ComputerSystem rather than to CIM_ComputerSystem unless the
130 context demands more clarity.

131 **1.2.1 Notation**

132 Augmented Backus-Naur Form (ABNF) [6] is used in this document to describe various aspects of
133 the SM ME addressing scheme. The complete grammar is provided in section 5.3.

134 The following conventions are used in this document:

- 135 • The – notation denotes the direction of containment and is read as “contains”. The
136 container is on the left of the arrow head.
- 137 • Specification elements in the `courier new` font indicates productions and literal
138 characters used in a grammar syntax expression.
- 139 • Specification elements in *italics* indicate a CIM association class in the body of this
140 document.

141 **1.3 Related Documents**

- 142 [1] Common Information Model (CIM) Schema, 2.14, <http://www.dmtf.org/standards/cim>
- 143 [2] *Unified Modeling Language (UML) from the Open Management Group (OMG)*,
144 <http://www.uml.org>
- 145 [3] [DSP0214](http://www.dmtf.org/standards/smash), *SM Command Line Protocol Specification*, 1.0.0,
146 <http://www.dmtf.org/standards/smash>
- 147 [4] [DSP2001](http://www.dmtf.org/standards/smash), *SM CLP Architecture White Paper*, 1.0.1, <http://www.dmtf.org/standards/smash>
- 148 [5] [DSP0217](http://www.dmtf.org/standards/smash), *SMASH Implementation Requirements*, 1.0.0, <http://www.dmtf.org/standards/smash>
- 149 [6] *Augmented BNF for Syntax Specifications: ABNF* IETF RFC 2234, November 1997,
150 <http://www.ietf.org/rfc/rfc2234.txt>

151

152 **1.4 Terminology**

Term	Definition
instance suffix	An instance suffix is a non-negative integer appended to the end of a UFcT to create an instance tag or UFiT. The instance suffix creates a unique identifier for the instance within the defining container. UFcT + instance suffix = UFiT
Addressing Association	SM ME Addressing defines certain CIM associations to be valid for the purpose of addressing CIM instances defined in SM profiles. These associations are based on the SM Containment Hierarchy. In the SM Address Grammar these associations are denoted by a slash (/) or a backslash (\) in the ME's UFiP. These associations are enumerated in Table 12 and Table 13.
Address Path	An Address Path is one in which each term has the appropriate intervening Addressing Association. (For more information, see section 4.3 and section 5.2.)

Term	Definition
Client	A Client is a logical component that manages a system through a Manageability Access Point (MAP). A Client may sometimes be referred to as the managing entity or management client.
Default UFcT	For a given class, this is the generic User-Friendly class Tag that is applicable for all instances of the class that are implemented.
Logical and Physical Containment	The SM ME Addressing scheme is based on the notion of object containment. In the logical CIM schema the ComputerSystem is the container for objects such as CIM Service, ServiceAccessPoint, LogicalDevice, and SystemSpecificCollection. In the physical CIM schema, the Rack is the top physical container. Table 12 and Table 13 describe the containers, the CIM relationship, and the instances they contain.
MAP address space	The hierarchical graph of the UFiTs contained in the MAP's AdminDomain. Each instance starting at the AdminDomain is a node in the graph. Each supported association forms a link in the graph to another instance node (and so on) until a terminating instance node is encountered.
Manageability Access Point (MAP)	A network-accessible interface for managing a Computer System. A MAP can be instantiated by a Management Process, a Management Processor, a Service Processor, or a Service Process.
Managed Element	An instance of CIM_ManagedElement
Managed Element Address	The UFiP to an ME instance. The UFiP may be used as a target of a SM CLP command.
Managed System address space	The fully connected graph of UFiTs that are contained in a specific managed system. The root instance is the ComputerSystem instance that represents the managed system.
Non-Addressing Association	A CIM Association class that is used in an SM profile but is not one of the SM Addressing Associations that are denoted by a slash (/) or backslash (\) in the UFiP. See section 5.3.
Root Instance	The top node in an SM address space; the MAP's AdminDomain
System Management Architecture for Server Hardware (SMASH)	An initiative to codify the management interface for heterogeneous servers in the data center, independent of machine state, operating system state, system topology, or access method
SM Command Line Protocol (CLP)	Defines a user-friendly command line protocol to manipulate CIM instances defined by DMTF Management Profiles

Term	Definition
SM Profile	A profile or profiles defined or referenced by <i>SMASH Implementation Requirements</i> . These specifications define CIM profiles for required and recommended instances, properties, and expected behavior to consistently model computer hardware platforms and management domains in order to achieve implementation interoperability.
Standard User-Friendly class Tags	The User-Friendly class Tags listed in the "UFcT" column of Table 1 through Table 11.
User-Friendly instance Path (UFiP)	A unique path to an instance formed by concatenating the UFiTs of each instance from the root instance to the terminating instance. The slash (/) or backslash (\) between each UFiT represents an Addressing Association.
User-Friendly Tag (UFT)	A short, user-friendly tag for a CIM class name or instance. The two types of UFTs are UFcT and UFiT (defined below).
User-Friendly class Tag (UFcT)	A short, user-friendly synonym for a CIM class name. It has the same properties and methods as the CIM class it represents.
User-Friendly instance Tag (UFiT)	A unique instance tag within the scope of the target instance's containment class. A UFiT is created by adding an non-negative integer suffix to the target instance's UFcT.

153 1.5 Acronyms and Abbreviations

Term	Definition
AA	Addressing Association
MAP	Manageability Access Point
ME	Managed Element
PE	Physical Element
SM	Server Management
SMASH	System Management Architecture for Server Hardware
SM CLP	Server Management Command Line Protocol
UFiP	User-Friendly instance Path
UFT	User-Friendly Tag
UFcT	User-Friendly class Tag
UFiT	User-Friendly instance Tag

154 **2 SM ME Addressing Overview**

155 This section describes Server Management Managed Element addressing.

156 **2.1 Goals**

157 This section enumerates the goals of the *SM ME Addressing Specification* and gives an overview of
158 how it achieves these goals.

159 The goals of the SM ME Addressing Specification are as follows:

- 160 • Provide an easy to use, user-friendly way to uniquely address CIM objects, using a
161 hierarchical containment structure based on specified CIM associations. This addressing
162 mechanism is intended to be applicable to other DMTF protocols.
- 163 • Provide access to information in other Managed Element (ME) instances associated with
164 the target ME instance through Non-Addressing Associations that are not part of a SM
165 Containment Hierarchy. This feature is required to support n dimensional association
166 traversal rooted at the terminating target UFiT. n is bounded by the Non-Addressing
167 Associations defined in the SM profiles supported by the MAP. Addressing Associations
168 are defined in section 4.3.
- 169 • Provide an unambiguous grammar to aid in programmatic parsing of a UFiP. The grammar
170 is defined in section 5.3.

171 **2.2 Conceptual Areas Addressed**

172 The following paragraphs discuss key conceptual areas that combine to achieve the goals described
173 in section 2.1.

174 CIM was chosen as the underlying data model because of its capacity to normalize computer-based
175 management relationships and information. One key aspect of the CIM schema is that it describes a
176 well-defined structured containment hierarchy that can be adapted to provide a non-ambiguous
177 method of addressing MEs.

178 User-Friendly class Tags (UFcT) are defined to simplify long, complex CIM class names. UFcTs are
179 simple, standard synonyms for the CIM classes used in the SM profiles. Rules are applied to UFcTs
180 to create User-Friendly instance Tags (UFiT). Other rules define how to form a unique User-
181 Friendly instance Path (UFiP). Section 3 defines User-Friendly class Tags, and section 5 defines the
182 rules for creating instance tags (UFiTs) based on the instance's class tag (UFcT).

183 The SM Containment Hierarchy described in section 4 defines the hierarchical CIM container-based
184 Addressing Associations that are combined to form a fully qualified address path (UFiP) to an
185 instance, much like the path to a file in a file system. The SM profiles define the specific CIM
186 objects that may be addressed and their supported Non-Addressing Associations. The SM CLP
187 Specification [3] defines how to use a SM ME Address in a CLP command and how to discover
188 what Non-Addressing Associations are supported for any given target ME.

189 3 User-Friendly Class Tags

190 A User-Friendly class Tag is designed to provide a short, user-friendly tag for a CIM class. A User-
191 Friendly class Tag is a simple substitute for a CIM class name. The UFcT is the basis for creating
192 User-Friendly instance Tags according to the containment shown in section 4. (Figure 1 and Figure 2
193 show the logical and physical containment relationships, respectively.) Section 5 describes how to
194 create a UFiT from a UFcT based on the instance's containment relationship.

195 Table 1 through Table 11 define standard User-Friendly class Tags.

196 The "CIM Class Name" column shows the CIM class name for each UFcT defined in the "UFcT"
197 column. CIM class names are organized according to the containment relationships defined in
198 Section 4. The term in bold in the UFcT column is the Default UFcT. The non-bold terms in this
199 column lists specific UFcTs that may be used instead of the Default UFcT.

200 Implementations can choose to offer optional OEM-specific UFcTs. Implementations prefix each
201 OEM-specific UFcT with the OEM key word followed by a vendor-unique identification string. The
202 OEM string portion of the prefix uniquely identifies the entity that owns and defines the OEM-
203 specific UFcT. The string includes a copyrighted, trademarked, or otherwise unique name that is
204 owned by the business entity or standards body defining the OEM-specific UFcT.

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

208 The "Based On" column gives an explanation of how the UFcT is determined. The *Class* value in
209 this column means that the UFcT is based on the CIM class name. If the UFcT is based on the value
210 of a CIM property within the class, the property name and value will be shown in the "Based On"
211 column. If the property is a string, the implementation interprets these strings in a case-insensitive
212 fashion. If the property is a value with a value map, the string provided is interpreted by the
213 implementation as the value map for the corresponding value of that property. Sometimes there is
214 more than one choice for a UFcT-to-Property value mapping. In these cases, the "Comments"
215 column provides some guidelines.

216 **Table 1 UFcTs for Subclasses of CIM_ManagedElement**

CIM Class Name	UFcT	Based On	Comments
CIM_Privilege	privilege	Class	
CIM_ConfigurationCapacity	configcapacity	Class	
CIM_LogEntry	record	Class	
CIM_UserContact	user	Class	
CIM_Identity	identity	Class	

Table 2 UFcTs for Subclasses of CIM_Capabilities

CIM Class Name	UFcT	Based On	Comments
CIM_BootServiceCapabilities	bootsvccap	Class	
CIM_CLPCapabilities	clpcap	Class	
CIM_EnabledLogicalElementCapabilities	elec	Class	
CIM_DeviceSharingCapabilities	sharingcap	Class	
CIM_PhysicalAssetCapabilities	phyassetcap	Class	
CIM_PowerManagementCapabilities	pwrmgtcap	Class	
CIM_SoftwareInstallationServiceCapabilities	swinstallsvccap	Class	
CIM_StorageCapabilities	storagecap	Class	
CIM_StorageConfigurationCapabilities	storagecfgcap	Class	
CIM_DHPCCapabilities	dhpcap	Class	
CIM_TelnetCapabilities	tnetcap	Class	
CIM_SSHCapabilities	sshcap	Class	
CIM_AccountManagementCapabilities	acctmgtcap	Class	
CIM_RoleBasedManagementCapabilities	rolemgtcap	Class	
CIM_ProcessorCapabilities	cpucap	Class	

Table 3 UFcTs for Subclasses of CIM_SettingData

CIM Class Name	UFcT	Based On	Comments
CIM_BootConfigSetting	bootcfgsetting	Class	
CIM_BootSourceSetting	bootsrcsetting	Class	
CIM_BootSettingData	bootsetting	Class	
CIM_DisplaySetting	displaysetting	Class	
CIM_StorageSetting	storagesetting	Class	
CIM_IPAssignmentSettingData	ipsettings	Class	
CIM_StaticIPAssignmentSettingData	staticipsettings	Class	
CIM_DHCPSettingData	dhcpsettings	Class	
CIM_DNSSettingData	dnssettings	Class	
CIM_DNSGeneralSettingData	dnsgeneralsettings	Class	
CIM_CLPSettingData	clpsetting	Class	
CIM_TelnetSettingData	telnetsetting	Class	
CIM_SSHSettingData	sshsetting	Class	
CIM_AccountSettingData	acctsetting	Class	

219

Table 4 UFcTs for Subclasses of CIM_Collection

CIM Class Name	UFcT	Based On	Comments
CIM_Group	group	Class	
CIM_RedundancySet	redundancysset	Class	
CIM_SystemSpecificCollection	availablesw	ElementName = "Available Software"	
CIM_ConcreteCollection	concretecollection	Class	
	hdwr	ElementName= "Hardware"	
	capabilities	ElementName = "Capabilities"	
	capacities	ElementName = "Capacities"	
	consoles	ElementName = "Consoles"	
	logs	ElementName = "Logs"	
	settings	ElementName = "Settings"	
CIM_Role	role	Class	

220

Table 5 UFcTs for Subclasses of CIM_LogicalElement

CIM Class Name	UFcT	Based On	Comments
CIM_SoftwareIdentity	swid	Class	
	swbundle	Classifications = "Software Bundle"	
CIM_StoragePool	storagepool	Class	

221

222

Table 6 UFcTs for Subclasses of CIM_EnabledLogicalElement

CIM Class Name	UFcT	Based On	Comments
CIM_Account	account	Class	
CIM_ConcreteJob	job	Class	
CIM_JobQueue	jobq	Class	
CIM_RecordLog	log	Class	
CIM_ProcessorCore	cpucore	Class	
CIM_HardwareThread	hwthread	Class	

Table 7 UFcTs for Subclasses of CIM_System

CIM Class Name	UFcT	Based On	Comments
CIM_AdminDomain	admin	Class	
	pwrdom	ElementName = "Power Domain"	
	coolingdom	ElementName = "Cooling Domain"	
CIM_ComputerSystem	system	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.
	router	Dedicated= "Gateway"	
	storagevlizer	Dedicated= "Storage Virtualizer"	
	medialib	Dedicated= "Media Library"	
	nashead	Dedicated= "NAS Head"	
	nas	Dedicated= "Self-contained NAS"	
	ups	Dedicated= "UPS"	
	iphone	Dedicated= "IP Phone"	
	map	Dedicated= "Manageability Access Point"	
	sp	Dedicated= "Management Controller"	
chassismgr	Dedicated= "Chassis Manager"		

Table 8 UFcTs for Subclasses of CIM_LogicalDevice

CIM Class Name	UFcT	Based On	Comments
CIM_Battery	battery	Class	
CIM_CDROMDrive	cd	Class	
CIM_CoolingDevice	cooling	Class	
CIM_DAPort	daport	Class	
CIM_DiskDrive	diskdrive	Class	
CIM_DisketteDrive	floppy	Class	
CIM_DiskPartition	diskpartition	Class	
CIM_Display	display	Class	
CIM_DVDDrive	dvd	Class	
CIM_Fan	fan	Class	
CIM_FCPort	fcport	Class	
CIM_HeatPipe	heatpipe	Class	
CIM_IBPort	ibport	Class	
CIM_Keyboard	keyboard	Class	
CIM_LogicalDisk	disk	Class	
CIM_LogicalModule	logicalmodule	Class	
	devicetray	LogicalModuleType = "Device Tray"	
	linecard	LogicalModuleType = "Line Card"	
	blademodule	LogicalModuleType = "Blade"	
CIM_LogicalPort	logicalport	Class	
CIM_MediaAccessDevice	mediaaccess	Class	
CIM_Memory	memory	Class	
CIM_Modem	modem	Class	
CIM_NetworkPort	netport	Class	
CIM_WirelessPort	wifiport	Class	
CIM_EthernetPort	enetport	Class	
CIM_PassThroughModule	passthru mod	Class	
CIM_PCIDevice	pcidev	Class	
CIM_PCIBridge	pcibrige	Class	
CIM_PointingDevice	pointer	Class	
	mouse	PointingType= "Mouse"	
	trackball	PointingType= "Track Ball"	
	touchpad	PointingType= "Touch Pad"	
	touchscreen	PointingType= "Touch Screen"	
CIM_PortController	portctrl	Class	
	nic	ControllerType= "Ethernet"	
	hca	ControllerType= "IB"	
	tca	ControllerType= "IB"	
	hba	ControllerType= "FC"	
CIM_PowerSupply	pwr supply	Class	
CIM_Printer	printer	Class	

CIM Class Name	UFcT	Based On	Comments
CIM_Processor	cpu	Class	
CIM_Refrigeration	refrigeration	Class	
CIM_SCSIProtocolController	scsiprotctrl	Class	
CIM_Sensor	sensor	Class	
CIM_NumericSensor	currentsensor	SensorType= "Current"	
	tachsensor	SensorType= "Tachometer"	
	tempsensor	SensorType= "Temperature"	
	voltsensor	SensorType= "Voltage"	
	countersensor	SensorType= "Counter"	
	switchsensor	SensorType= "Switch"	
	locksensor	SensorType= "Lock"	
	humiditysensor	SensorType= "Humidity"	
	airsensor	SensorType= "Air Flow"	
	presencesensor	SensorType= "Presence"	
	smokesensor	SensorType= "Smoke Detection"	
CIM_SPIPort	spiport	Class	
CIM_StorageVolume	storagevol	Class	
CIM_StorageExtent	storageext	Class	
CIM_SerialPort	serialport	Class	
CIM_TapeDrive	tapedrive	Class	
CIM_USBPort	usbport	Class	
CIM_WatchDog	watchdog	Class	

225

Table 9 UFcTs for Subclasses of CIM_Service

CIM Class Name	UFcT	Based On	Comments
CIM_BootService	bootsvc	Class	
CIM_IPConfigurationService	ipcfgsvc	Class	
CIM_PowerManagementService	pwrmtgsvc	Class	
CIM_SharedDeviceManagementService	shareddevicesvc	Class	
CIM_SoftwareInstallationService	swinstallsvc	Class	
CIM_StorageConfigurationService	storagecfgsvc	Class	
CIM_TextRedirectionService	textredirectsvc	Class	
CIM_TimeService	timesvc	Class	
CIM_ProtocolService	protosvc	Class	
	sshsvc	Class	
	telnetsvc	Class	
	clpsvc	Class	
CIM_NetworkPortConfigurationService	netportcfgsvc	Class	
CIM_RoleBasedAuthorizationService	rolesvc	Class	
CIM_AccountManagementService	acctsvc	Class	

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	
	dnserver	AccessContext="DNS Server"	
	dhcpserver	AccessContext="DHCP Server"	
	gateway	AccessContext="Default Gateway"	
CIM_LANEndpoint	lanendpt	Class	
CIM_SCSIProtocolEndpoint	scsiendpt	Class	
CIM_TextRedirectionSAP	textredirectsap	Class	
CIM_TelnetProtocolEndpoint	telnetendpt	Class	
CIM_CLPPProtocolEndpoint	clpendpt	Class	
CIM_TCIPProtocolEndpoint	tcpendpt	Class	
CIM_SSHProtocolEndpoint	sshendpt	Class	
CIM_SoftwareIdentityResource	swidres	Class	

Table 11 UFcTs for Subclasses of CIM_PhysicalElement

CIM Class Name	UFcT	Based On	Comments
CIM_PhysicalPackage	pkg	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	frame	Class	

CIM Class Name	UFcT	Based On	Comments
CIM_Rack	rack	Class	
CIM_Chassis	chassis	Class	
	laptop	ChassisPackageType= "LapTop"	
	desktop	ChassisPackageType= "Desktop"	
	tower	ChassisPackageType= "Tower"	
	storagechassis	ChassisPackageType= "Storage Chassis"	
	notebook	ChassisPackageType= "Notebook"	
	mainchassis	ChassisPackageType= "Main System Chassis"	
	expansion	ChassisPackageType= "Bus Expansion Chassis"	
	peripheralchassis	ChassisPackageType= "Peripheral Chassis"	
subchassis	ChassisPackageType= "SubChassis"		
CIM_Card	card	Class	
CIM_PhysicalComponent	component	Class	
CIM_Chip	chip	Class	
	propchip	FormFactor= "Proprietary Chip"	
	sip	FormFactor= "SIP"	
	dip	FormFactor= "DIP"	
	zip	FormFactor= "ZIP"	
	soj	FormFactor= "SOJ"	
	simm	FormFactor= "SIMM"	
	dimm	FormFactor= "DIMM"	
	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 Class Name	UFcT	Based On	Comments
CIM_PhysicalMemory	pmem	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"	
	eprom	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	connector	class	
CIM_Slot	slot	class	

228 4 SM Containment Hierarchy

229 This section describes the SM CIM-based containment hierarchy that forms the foundation of the
230 SM ME Addressing Standard. The Physical Containment Hierarchy defines how PhysicalElements
231 are addressed. The Logical Containment Hierarchy defines how LogicalElements are addressed. The
232 SM Containment Hierarchy is based on CIM Schema V2.14 and will be revised to accommodate
233 later CIM Schema versions as required.

234 CIM divides its schema into two realms composed of logical and physical objects. All logical objects
235 derive from LogicalElement. All physical objects derive from PhysicalElement. In CIM, all aspects
236 of an object that are not related to its three dimensional concrete real world existence are modeled as
237 LogicalElements. This includes devices. A CIM_PhysicalElement (PE) is an object that occupies
238 space and can be touched. A LogicalElement, on the other hand, cannot. This seems a bit non-
239 intuitive for some LogicalElements like ComputerSystem and LogicalDevice. The ComputerSystem
240 is the LogicalElement that comprises all the functionality of a system, not just the hardware, but
241 software, firmware, processes, file systems, internet addresses, logs, etc. In a single chassis system
242 one can imagine one is touching the entire computer system, but in reality one is only touching the
243 form factor that encloses the computer. Similarly, one may argue that a device is a tangible thing.
244 One can touch a disk drive. As in the previous example, the disk drive is the *physical* form factor (a
245 PhysicalPackage) that encloses the *logical* aspects of the disk, such as storage volumes, disk
246 partitions, and memory blocks. In CIM, only the form factor is a PhysicalPackage and all the rest,
247 including the logical entity that describes the kind of disk drive (floppy, CD, or DVD), are
248 LogicalDevices.

249 The SM ME Addressing scheme is based on the hierarchical pattern of associations between CIM
250 objects. A UFiP is created by identifying the specific object (class or instance) tags (UFITs) to a leaf
251 object starting from the root object (AdminDomain). SM ME Addressing uses a slash (/) or a
252 backslash (\) between object terms to signify that an Addressing Association (AA) exists between
253 the terms. The SM Containment Hierarchies define the Addressing Associations. The Addressing
254 Associations are described in section 4.4, and the SM ME Addressing Containment relationships are
255 listed in Table 12 and Table 13. These relationships are stated more formally in section 5.3.

256 Another key concept is the difference between a class and an instance of a class. A class is the
257 template one uses to create an instance. The class defines the nature of the properties and methods;
258 data type, size, range, signature (input, output), etc. An instance is a unique instantiation of the class
259 template with specific information for each variable term. So, when one uses a class tag (UFcT),
260 such as cpu or disk, one is addressing a particular class. When one uses an instance tag (UFiT), one
261 is addressing a particular object, such as cpu1 or disk3, which contains specific information for one
262 particular CPU or disk.

263 4.1 Logical Containment

264 The root of the SM MAP address space is the AdminDomain. The term *root* denotes the highest
265 order container in which the MAP stores information. Each class in the hierarchy is considered to be
266 a container for its own information, including association classes that contain information regarding
267 the classes they associate.

268 The ComputerSystem is the root of its address space. That is, all the information about the Managed
269 Elements within a single computer system is contained within the ComputerSystem through
270 associations to the contained instance or instances.

271 The UML diagram in Figure 1 defines the SM Logical Containment hierarchy. At the top of the
272 diagram is the AdminDomain. For each ComputerSystem managed by a MAP, there is a
273 *<SystemComponent>* association between the AdminDomain and the ComputerSystem. From here
274 the Client can traverse intervening Addressing Associations to address SM MEs. The parser
275 validates that each UFiP is valid according to the grammar. This is discussed in more detail in
276 section 5.2.

277 The diagram in Figure 1 illustrates that the SM Logical Containment Hierarchy provides the
278 following associations to address logical entities within a ComputerSystem and define their
279 containment relationship. The ← notation denotes the direction of containment. The container is on
280 the left of the arrow head and is read as “contains”. The Addressing Association is shown in angle
281 brackets and italics. Note that the following list of associations is not exhaustive, nor are the
282 relationships shown in Figure 1, Figure 2, and Figure 3 exhaustive. The complete list of allowable
283 managed elements is shown in the tables and in section 5.

284 ComputerSystem ← *<HostedService>* ← Service

285 ComputerSystem ← *<HostedAccessPoint>* ← ServiceAccessPoint

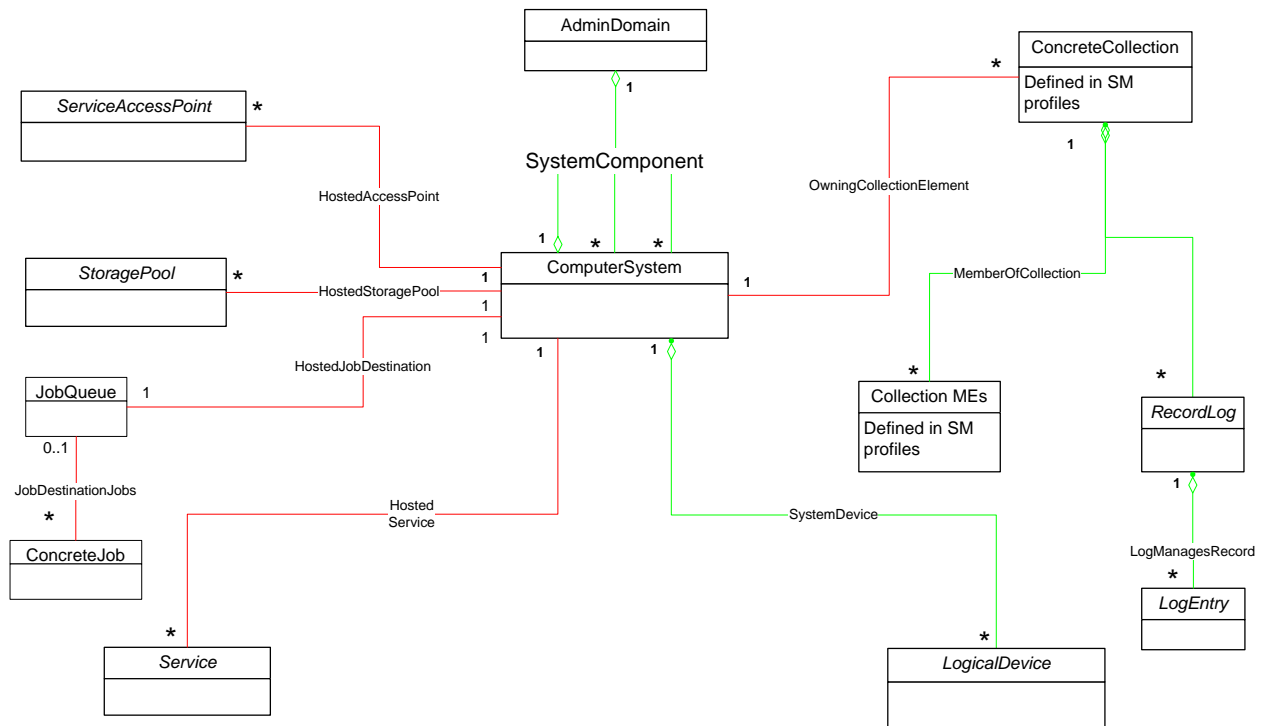
286 ComputerSystem ← *<OwningCollectionElement>* ← ConcreteCollection

287 ComputerSystem ← *<HostedStoragePool>* ← StoragePool

288 ComputerSystem ← *<SystemComponent>* ← ComputerSystem

289 ComputerSystem ← *<SystemDevice>* ← LogicalDevice

290 Note that the cardinality in the CIM Schema may be different than what is represented in Figure 1,
291 Figure 2, and Figure 3. To enforce containment, this specification restricts the cardinality of the
292 aggregations to what is shown in the figures.



293

294

Figure 1 SM Logical Containment UML Diagram

295 The relationships described in the previous paragraph can be expressed in the following hierarchical
 296 notation:

- 297 AdminDomain/ComputerSystem
- 298 AdminDomain/ComputerSystem/Service
- 299 AdminDomain/ComputerSystem/ServiceAccessPoint
- 300 AdminDomain/ComputerSystem/ConcreteCollection
- 301 AdminDomain/ComputerSystem/ConcreteCollection/ME
- 302 AdminDomain/ComputerSystem/ConcreteCollection/RecordLog
- 303 AdminDomain/ComputerSystem/ConcreteCollection/RecordLog/LogEntry
- 304 AdminDomain/ComputerSystem/LogicalDevice
- 305 AdminDomain/ComputerSystem/ComputerSystem/LogicalDevice

306 In the above notation, a slash (/) or backslash (\) is a shorthand notation for the intervening
 307 Addressing Association. Section 5.1 describes how a UFiT is created from a UFcT. The resultant
 308 UFiTs can then be substituted for the CIM class names in the previous list to form distinct SM ME
 309 instance addresses. Multiple Addressing Associations may associate a ME to its containers. The
 310 implementation selects exactly one of these AAs to use for addressing all instances of the ME.

311 In Figure 1, CIM defines that a single AdminDomain may contain 0 to n ComputerSystem instances
312 and 0 to n Collection instances. A ComputerSystem contained in the AdminDomain may contain

- 313 • 0 to n Service instances
- 314 • 0 to n ServiceAccessPoint instances
- 315 • 0 to n Collection instances
- 316 • 0 to n LogicalDevice instances
- 317 • 0 to n StoragePool instances
- 318 • 0 to n contained ComputerSystem instances that also may contain 0 to n LogicalDevice
319 instances, Service instances, and so on.

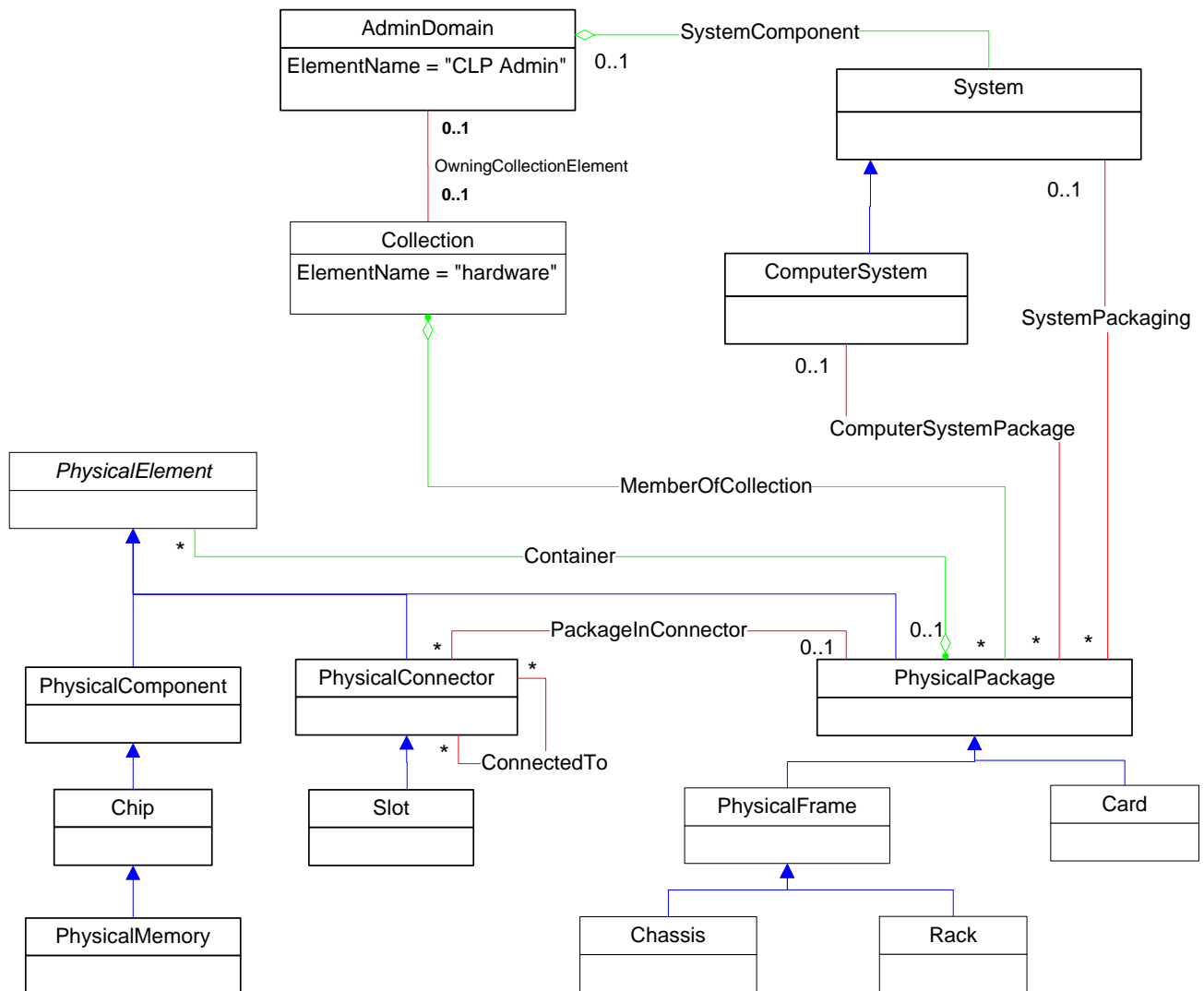
320 This observation forms the basis of the addressing rules defined in section 5.

321 ConcreteCollection represents all the subclasses of that class that appear in the SM profiles,
322 including SystemSpecificCollection, RedundancySet, Group, and ConcreteCollection. The term
323 “Collection MEs” associated to the Collection through MemberOfCollection or
324 OrderedMemberOfCollection represents the MEs that are contained in the Collection (or a subclass)
325 instance.

326 Similarly, ComputerSystem, ServiceAccessPoint, LogicalDevice, and Service represent all the
327 subclasses of these classes that appear in the SM Profiles.

328 **4.2 Physical Containment**

329 The example in Figure 1 shows the SM Physical Containment Hierarchy and the associations that
 330 define the SM ME Addressing scheme and the containment associations for server hardware
 331 managed in the MAP's AdminDomain.



332

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

334 CIM defines several aggregation associations for PEs. SM addressing leverages the rules of the
 335 associations defined in Figure 2 to uniquely address any PE. Figure 2 shows that the MAP
 336 AdminDomain contains a single ConcreteCollection with the property ElementName set to
 337 "hardware". Through *MemberOfCollection*, this collection contains the topmost PhysicalPackage
 338 instance of a ComputerSystem that the MAP manages. If this is a Rack object, then it is the top of
 339 the hierarchy that contains all the PEs in that Rack. A Chassis within this Rack would be related with
 340 a *Container* association between the contained Chassis (as the PartComponent reference) and the
 341 Rack container (as the GroupComponent reference). Similarly, the PhysicalPackages in each
 342 Chassis would be related with a *Container* association, where the container Chassis is the
 343 GroupComponent and the PhysicalPackage is the PartComponent. PhysicalPackages in turn may

344 contain any type of PhysicalElement (including other PhysicalPackages). The container
345 PhysicalPackage is always defined using the GroupComponent reference, and the contained
346 PhysicalElement is defined to be the PartComponent.

347 A ComputerSystem may be composed of hardware components that are contained in a single form
348 factor (Chassis) or components that span multiple Racks. Figure 2 shows that the ComputerSystem is
349 associated to instances of its topmost PhysicalPackage instance through the
350 <ComputerSystemPackage> association, and a System is associated to its topmost PhysicalPackage
351 instance through the <SystemPackaging> association.

352 In Figure 2, the PhysicalElement inheritance hierarchy illustrated by the blue arrows shows that a PE
353 may be a PhysicalPackage (frame, rack, and chassis), Component, Connector, or PhysicalLink.
354 Because a Chassis is a PhysicalPackage, it may be substituted for a PhysicalPackage in the Physical
355 Containment diagram in Figure 2, adding *n* levels of recursion. A PhysicalPackage may also contain
356 any PE, adding another level of recursion. Thus, the potential hardware containment relationship
357 hierarchy is *n* levels deep, which is complex enough to model any arbitrary computer hardware
358 topology. The SM profiles provide guidelines to implementers on how to model hardware platforms
359 to conform to the containment hierarchy. Vendors decide which components in their systems will be
360 manageable by creating the appropriate ME instances. The rules for ME containment are described
361 in sections 4.3 and 5.

362 The Physical Containment diagram in Figure 2 shows the MAP AdminDomain top down
363 containment relationships as follows:

364 AdminDomain ← <OwningCollectionElement> ← ConcreteCollection ← <MemberOfCollection> ←
365 PhysicalPackage

366 Rack ← <Container> ← Chassis ← <Container> ← PhysicalPackage

367 ← <Container> ← PhysicalComponent

368 or

369 ← <Container> ← PhysicalConnector

370 or

371 ← <Container> ← PhysicalElement

372 or

373 Chassis ← <Container> ← PhysicalPackage

374 ← <Container> ← PhysicalComponent

375 or

376 ← <Container> ← PhysicalConnector

377 or

378 ← <Container> ← PhysicalElement

379 or

380 PhysicalPackage ← <Container> ← PhysicalComponent

381 or

382 ← <Container> ← PhysicalConnector

383 or

384 ← <Container> ← PhysicalElement

385 By definition the MAP AdminDomain contains a single ConcreteCollection that contains 0 to n
386 PhysicalPackage instances (Rack, Chassis, or PhysicalPackage) that represent the topmost physical
387 containers of the ComputerSystem instances that the MAP is managing. A single Rack contains 0 to
388 n Chassis. A single Chassis contains 0 to n PhysicalPackage instances. A single PhysicalPackage
389 contains 0 to n PhysicalComponent, PhysicalConnector, or PhysicalElement instances.

390 The SM containment rules in section 5 guarantee that contained elements are unique. The SM
391 Physical Containment Hierarchy associations prescribe how containers are ordered. The specific
392 association is never ambiguous because it is explicit in the instance according to the reference role
393 selected.

394 The MAP AdminDomain Addressing Associations and containment relationships are expressed in
395 the following notation, where the slash (/) or backslash (\) denotes the intervening association
396 defined in Figure 2:

397 AdminDomain/ConcreteCollection/Rack/Chassis/PhysicalPackage

398 AdminDomain/ConcreteCollection/Rack/Chassis/PhysicalPackage/PhysicalConnector

399 AdminDomain/ConcreteCollection/Rack/Chassis/PhysicalPackage/PhysicalComponent

400 AdminDomain/ConcreteCollection/Rack/Chassis/PhysicalPackage/PhysicalElement

401 AdminDomain/ConcreteCollection/Chassis/PhysicalPackage

402 AdminDomain/ConcreteCollection/Chassis/PhysicalPackage/PhysicalConnector

403 AdminDomain/ConcreteCollection/Chassis/PhysicalPackage/PhysicalComponent

404 AdminDomain/ConcreteCollection/Chassis/PhysicalPackage/PhysicalElement

405 AdminDomain/ConcreteCollection/PhysicalPackage

406 AdminDomain/ConcreteCollection/PhysicalPackage/PhysicalConnector

407 AdminDomain/ConcreteCollection/PhysicalPackage/PhysicalComponent

408 AdminDomain/ConcreteCollection/PhysicalPackage/PhysicalElement

409 Note that the Physical Containment Hierarchy can be arbitrarily deep based on the hardware system
410 that is being represented.

411 An implementation may choose not to implement instances of every PE class; therefore, in those
412 implementations, some terms in the address path may not appear. At least one of the following
413 associations is required, depending on the type of the topmost PhysicalPackage being managed:

414 AdminDomain/ConcreteCollection/Rack

415 AdminDomain/ConcreteCollection/Chassis

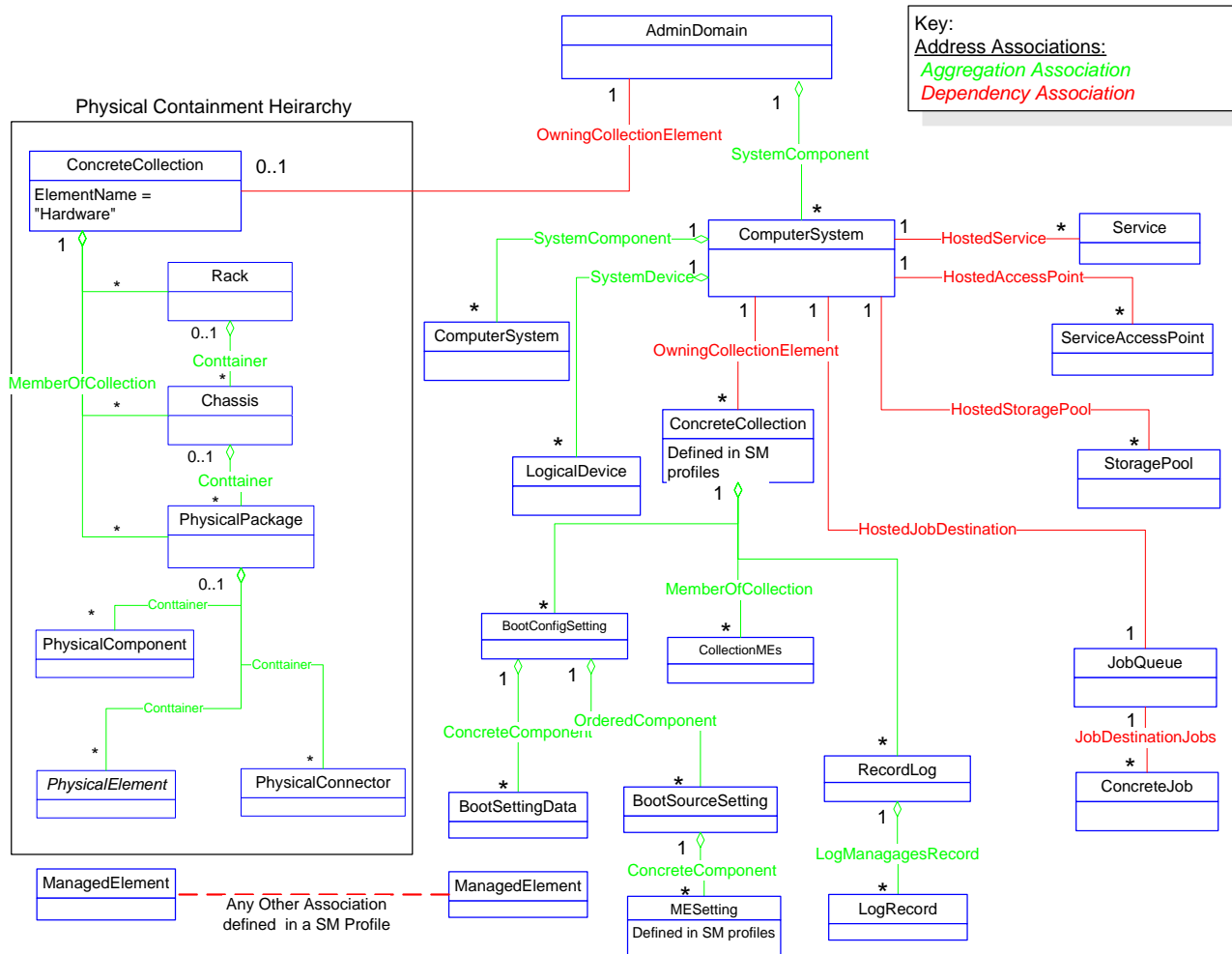
416 AdminDomain/ConcreteCollection/PhysicalPackage

417 MAP implementations define which PEs are available to be managed by creating instances of the PE
418 according to the SM profiles. Section 5.1 describes how a UFiT is created from a UFcT. The
419 resultant UFiTs can then be substituted for the CIM class names in the Physical Containment
420 Hierarchy to form distinct SM PE instance addresses.

421 **4.3 SM ME Addressing Hierarchy**

422 In summary, the SM Logical Hierarchy uses CIM logical containment associations to address
 423 Services, ServiceAccessPoints, Collections, ComputerSystems, StoragePools, and LogicalDevices in
 424 a ComputerSystem in the MAP's AdminDomain. The SM Physical Hierarchy defines how to
 425 address physical components in the MAP's AdminDomain. The rules in section 5.2 define how to
 426 create a UFiT from a UFcT so that it is unique within the managed ComputerSystem.

427 The diagram in Figure 3 combines the components of the Logical and Physical Containment
 428 Hierarchies to define the SM ME Addressing Hierarchy.



429

430 **Figure 3 High Level View SM ME Address Hierarchy**

431 Figure 3 shows a high level example of the SM ME Addressing Hierarchy, which is not intended to
 432 be an exhaustive list of Addressing Associations. The SM ME Addressing Grammar in section 5.3
 433 presents all the valid addressing terms. Figure 3 shows a single hierarchical address space rooted at
 434 the AdminDomain. An Address Path is one in which each term has the appropriate intervening
 435 Addressing Association and role. This is explained in detail in the rules in section 5.2. The diagram
 436 shows the Addressing Associations between the path terms that must exist at each level until the last
 437 (leaf) term is reached.

438 Table 12 and Table 13 give examples of the Addressing Associations that can be used to form legal
 439 addresses. This is not a complete list. The “Addressing Association Class” column lists the CIM
 440 association classes that may be used to form a legal address between instances of the concrete class
 441 (and its subclasses) listed in the “Container Class” column and instances of the class (and its
 442 subclasses) listed in the “Contained Class” column. The grammar described in section 5 defines how
 443 the legal association class is expressed as a slash (/) when forming an instance path.

444 Table 12 lists the SM physical containment associations based on the Physical Containment
 445 Hierarchy. The class’s role in the association is used to determine its position in the address path.

446 **Table 12 SM Hardware Containment Relationships**

Container Class (REF Role)	Addressing Association Class	Contained Class (REF Role)
CIM_AdminDomain (OwningElement)	CIM_OwningCollectionElement	CIM_ConcreteCollection (OwnedElement)
CIM_ConcreteCollection (Collection)	CIM_MemberOfCollection	CIM_PhysicalPackage (Member)
CIM_Rack (GroupComponent)	CIM_Container	CIM_Chassis (PartComponent)
CIM_Chassis (GroupComponent)	CIM_Container	CIM_PhysicalPackage (PartComponent)
CIM_PhysicalPackage (GroupComponent)	CIM_Container	CIM_PhysicalElement (PartComponent)
CIM_PhysicalPackage (GroupComponent)	CIM_Container	CIM_PhysicalComponent (PartComponent)
CIM_PhysicalPackage (GroupComponent)	CIM_Container	CIM_PhysicalConnector (PartComponent)
CIM_Card (GroupComponent)	CIM_Container	CIM_Card (PartComponent)
CIM_PhysicalConnector (Antecedent)	CIM_PackageInConnector	CIM_PhysicalPackage (Dependent)
CIM_Slot (Antecedent)	CIM_PackageInConnector	CIM_PhysicalPackage (Dependent)
CIM_Slot (Antecedent)	CIM_PackageInConnector	CIM_Card (Dependent)
CIM_PhysicalConnector (Antecedent)	CIM_ConnectedTo	CIM_PhysicalConnector (Dependent)

447 Table 13 lists the SM logical containment associations based on the Logical Containment Hierarchy.
 448 These associations are used in section 5 to derive unique instance tags for SM logical MEs. The
 449 class's role in the association is used to determine its position in the address path.

450

Table 13 SM Logical Containment Relationships

Container Class (REF Role)	Addressing Association Class	Contained Class (REF Role)
CIM_AdminDomain (GroupComponent)	CIM_SystemComponent	CIM_ComputerSystem (PartComponent)
CIM_AdminDomain (OwningElement)	CIM_OwningCollectionElement	CIM_ConcreteCollection (OwnedElement)
CIM_ComputerSystem (GroupComponent)	CIM_SystemComponent	CIM_System (PartComponent)
CIM_ComputerSystem (GroupComponent)	CIM_SystemDevice	CIM_LogicalDevice (PartComponent)
CIM_ComputerSystem (Antecedent)	CIM_HostedService	CIM_Service (Dependent)
CIM_ComputerSystem (Antecedent)	CIM_HostedAccessPoint	CIM_ServiceAccessPoint (Dependent)
CIM_ComputerSystem (OwningElement)	CIM_OwningCollectionElement	CIM_Collection (OwnedElement)
CIM_ComputerSystem (Antecedent)	CIM_HostedCollection	CIM_SystemSpecificCollection (Dependent)
CIM_ComputerSystem (GroupComponent)	CIM_HostedStoragePool	CIM_StoragePool (PartComponent)
CIM_ProtocolService (Antecedent)	ProvidesEndpoint	CIM_ProtocolEndpoint (Dependent)
CIM_ProtocolService (Antecedent)	ServiceAccessBySAP	CIM_ProtocolEndpoint (Dependent)
CIM_ConcreteCollection (Collection)	CIM_MemberOfCollection	CIM_ManagedElement (Member)
CIM_SystemSpecificCollection (Collection)	CIM_MemberOfCollection	CIM_SoftwareIdentity (Member)
CIM_ComputerSystem (System)	CIM_InstalledSoftwareIdentity	CIM_SoftwareIdentity (SoftwareIdentity)
CIM_BootConfigSetting (GroupComponent)	CIM_ConcreteComponent	CIM_SettingData (PartComponent)
CIM_BootConfigSetting (GroupComponent)	CIM_OrderedComponent	CIM_BootSourceSetting (PartComponent)
CIM_BootSourceSetting (GroupComponent)	CIM_ConcreteComponent	CIM_SettingData (PartComponent)
CIM_BootSourceSetting (SystemElement)	CIM_LogicalIdentity	CIM_BootConfigSetting (SameElement)
CIM_RecordLog (Log)	CIM_LogManagesRecord	CIM_LogEntry (Record)
CIM_ComputerSystem (Antecedent)	CIM_HostedJobDestination	CIM_JobQueue (Dependent)
CIM_JobQueue (Antecedent)	CIM_JobDestinationJobs	CIM_ConcreteJob (Dependent)
CIM_ProtocolEndpoint (Antecedent)	CIM_BindsTo	CIM_ServiceAccessPoint (Dependent)

Container Class (REF Role)	Addressing Association Class	Contained Class (REF Role)
CIM_ProtocolEndpoint (Antecedent)	CIM_SAPSAPDependency	CIM_ProtocolEndpoint (Dependent)
CIM_ProtocolEndpoint (Dependent)	CIM_RemoteAccessAvailableToElement	CIM_RemoteServiceAccessPoint (Antecedent)
CIM_EthernetPort (Antecedent)	CIM_DeviceSAPImplementation	CIM_LANEEndpoint (Dependent)
CIM_IPAssignmentSettingData (GroupComponent)	CIM_OrderedComponent	CIM_IPAssignmentSettingData (PartComponent)
CIM_LogicalModule (GroupComponent)	CIM_ConcreteComponent	CIM_LogicalDevice (PartComponent)
CIM_Processor (GroupComponent)	CIM_ConcreteComponent	CIM_ProcessorCore (PartComponent)
CIM_ProcessorCore (GroupComponent)	CIM_ConcreteComponent	CIM_HardwareThread (PartComponent)
CIM_ComputerSystem	CIM_AccountOnSystem	CIM_Account
CIM_Account (ManagedElement)	CIM_AssignedIdentity	CIM_Identity (Identity)
CIM_Group (ManagedElement)	CIM_AssignedIdentity	CIM_Identity (Identity)
CIM_Identity (Identity)	CIM_AssignedIdentity	CIM_UserContact (ManagedElement)
CIM_AccountManagementService (Service)	CIM_ServiceAffectsElement	CIM_Identity (ManagedElement)
CIM_RoleBasedAuthorizationService (Antecedent)	CIM_ConcreteDependency	CIM_Privilege (Dependent)
CIM_ComputerSystem (ManagedElement)	CIM_SAPAvailableForElement	CIM_TextRedirectionSAP (AvailableSAP)
CIM_SoftwareIdentity (Dependent)	CIM_OrderedDependency	CIM_SoftwareIdentity (Antecedent)
CIM_SoftwareIdentity (GroupComponent)	CIM_OrderedComponent	CIM_SoftwareIdentity (PartComponent)
CIM_SoftwareIdentity (ManagedElement)	CIM_SAPAvailableForElement	CIM_SoftwareIdentityResource (ServiceAccessPoint)

451 **4.4 Addressing Associations**

452 This specification not only defines a way of addressing instances of classes, but it also defines a way
453 of addressing instances of associations. The SM ME Addressing Grammar defines the following
454 syntax to address CIM Association instances:

455 UFiP=> <TargetAssoc>

456 UFiP=> <TargetAssoc> =>UFiP

457 The first form targets all associations of type <TargetAssoc> that reference the instance specified in
458 the preceding UFiP, and the second form targets the Association instance that links the UFiP on
459 each side of the expression.

460 The following example, which uses the details provided in Figure 1, may help explain the use of this
461 feature:

462 Admin Domain ← <SystemComponent> ←ComputerSystem ← <SystemDevice> ←
463 LogicalDevice

464 Substituting the slashes for the associations results in

465 AdminDomain/ComputerSystem/LogicalDevice

466 If the LogicalDevice were a Processor, a possible UFiP for this example could be

467 admin1/system1/cpu1

468 The following syntax could be used to address all instances of the SystemDevice Association that
469 were associated with this instance of AdminDomain/ComputerSystem:

470 admin1/system1=>systemdevice

471 The specific instance of SystemDevice that is associated with this instance of the LogicalDevice
472 under this AdminDomain/ComputerSystem could be addressed as follows:

473 admin1/system1=>systemdevice=>admin1/system1/cpu1

474 **5 SM ME Addressing Rules**

475 This section details the normative requirements for implementing support for the SM ME
 476 Addressing specification. Section 5.1 describes how User-Friendly instance Tags are created.
 477 Section 5.2 defines the rules that implementations MUST adhere to in order to comply with the SM
 478 ME Addressing specification. Section 5.3 shows the productions for the SM ME Addressing
 479 Grammar. This grammar provides the basis to develop algorithms to programmatically parse a UFiP.

480 **5.1 Instance Tagging**

481 The User-Friendly instance Tag (UFiT) is constructed by adding an instance suffix to the UFcT
 482 (**default** or **specific**). The instance suffix is a non-negative integer and is unique within its container
 483 for any given UFcT. It is recommended that suffixes within a container be numbered starting at one
 484 and increase in value sequentially.

485 A unique UFiT within a container addresses a single ME instance. Appendix A shows several
 486 examples of how to create SM UFiTs and UFiPs based on an instance’s UFcT and container.

487 **5.2 Addressing Rules**

488 This section defines the rules that implementations MUST obey to support SM ME Addressing.

489 **Table 14 SM ME Addressing Rules**

Rule	Description
1	Implementations MUST implement User-Friendly class Tags as follows: <ul style="list-style-type: none"> • Implementations MUST implement the default UFcT for each instance of a class that is implemented as defined in the profile being implemented. • Implementations MUST derive the UFcT from the implemented class. • Implementations MUST use the Standard UFcTs from Table 1 through Table 11 for the class instantiated. • Implementations MAY use any of the specific UFcTs listed in Table 1 through Table 11 for the class instantiated. • If an implementation uses the default UFcT, then the property that defines the specific UFcT MAY be null. • If an implementation uses a specific UFcT based on a property, that property value MUST NOT be null and MUST agree with the property value listed for the specific UFcT in the “Based On” column of the Standard User-Friendly class Tags table. If the property is a string, the implementation MUST interpret the value of the property in a case-insensitive fashion. If the property is a value mapped property, the interpretation MUST interpret the string specified as the ValueMap for the corresponding value. • 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 MUST use the value of the first element in the array as the selector for the optional, specific UFcT. • Implementations MUST interpret all UFcTs and UFiTs in a case-insensitive fashion. • When a UFcT is the default for a CIM class, the implementation MUST respond to a query for all instances of the UFcT with all instances of that class that could have had the UFcT assigned. Implementations MAY return instances of a subclass when the subclass is not defined in the DMTF Core or Common Schema. Implementations SHOULD NOT return instances of a subclass when the subclass is defined in the DMTF Core or Common Schema.

Rule	Description
2	<p>Implementations MUST support the following rules for creating instance tags from SM UFcT the class root and container class defined in section 5.1:</p> <ul style="list-style-type: none"> a. The UFiT MUST be constructed by adding an instance suffix to the UFcT (default or specific). b. The instance suffix MUST be a non-negative integer. c. The instance suffix MUST be unique within its container for any given UFcT. d. Instance suffixes are not required to be a sequential enumeration. e. Instance suffixes are not required to begin at any given value for any container.
3	<p>Implementations' ME addresses MUST comply with the syntax specified in section 5.3.</p>
4	<p>Implementations MUST define ME addresses according to the Addressing Associations defined in the SM Containment Hierarchy and SM profiles:</p> <ul style="list-style-type: none"> • Implementations MUST provide exactly one UFiP from the AdminDomain to every ME. • Implementations MUST guarantee that a UFiP identifies exactly one ME instance. • Implementations MUST choose at most one Addressing Association to address an instance of an ME. • Implementations MUST 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. • Implementations MUST guarantee that the UFiT assigned to the instance in the container role precedes the UFiT assigned to the instance in the contained role as defined in Table 12 and Table 13.
5	<p>Implementations MUST NOT allow UFiTs to be modified by the Client.</p>
6	<p>Implementations MAY provide OEM-specific extensions to the Standard User-Friendly class Tags. The meaning of any UFcT with an OEM prefix is outside the scope of this specification. Implementations that support OEM-specific extensions MUST adhere to the following rules:</p> <ul style="list-style-type: none"> • Implementations MUST prefix each OEM specific UFcT with the key word OEM followed by a vendor-unique identification string. • The OEM string portion of the prefix MUST uniquely identify the entity that owns and defines the OEM-specific UFcT. • The string MUST include a copyrighted, trademarked, or otherwise unique name that is owned by the business entity or standards body defining the OEM-specific UFcT.
7	<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 that do so MUST adhere to the following rules:</p> <ul style="list-style-type: none"> • Implementations MUST define the conditions under which UFiT are destroyed. • Implementations MUST define the algorithm, if any, that is used to recycle a UFiT.

490 **5.2.1 Rules for Selecting Instances by UFcT**

491 The Selection UFcT is the UFcT for which matches are searched. The following are the rules for
492 selecting instances by UFcT. If any of these conditions are met, the instance MUST be selected by
493 the Selection UFcT.

- 494 • if the Selection UFcT is a Default UFcT and the class instantiated is the class identified in
495 the "CIM Class Name" column for the Selection UFcT specified in the "UFcT" column in
496 one of the UFcT tables listed in User-Friendly Class Tags (section 3).
- 497 • if the Selection UFcT is the UFcT that has been appended with an instance suffix to create
498 the UFiT of the instance

499 The following examples illustrate successful matches:

- 500 • Selection UFcT = system , UFiT = switch1

501 This is a successful match because system is the default UFcT for the
502 CIM_ComputerSystem class. An instance with a UFiT of switch1 is an instance of
503 CIM_ComputerSystem.

- 504 • Selection UFcT = switch, UFiT = switch1

505 This is a successful match because switch is the UFcT used to produce the UFiT switch1.

506 The following examples illustrate unsuccessful matches:

- 507 • Selection UFcT = sensor, UFiT = numsensor1

508 This match is unsuccessful because the instantiated class is CIM_NumericSensor and not
509 CIM_Sensor.

- 510 • Selection UFcT = mouse, UFiT = trackball1

511 In this case the instantiated class is CIM_PointingDevice, which has a default UFcT of
512 pointer. This match is unsuccessful because mouse is not the default UFcT for the
513 instantiated instance, nor is it the UFcT used to construct the UFiT assigned to the instance.

514 **5.3 SM ME Addressing Grammar**

515 This section describes the grammar productions needed to form valid SM ME addresses. The
516 production rules and notation are defined in [RFC2234](#) [6]. Implementations MUST expose ME
517 addresses that comply with the MEAddress production specified by the SM ME Address Grammar
518 in this section. Implementations MUST use the UFcT production to address a CIM class. IETF
519 [RFC2234](#) defines a modified version of Backus-Naur Form (BNF), called Augmented BNF (ABNF),
520 which is used in many Internet specifications. It balances compactness and simplicity, and it
521 provides reasonable representational power. [RFC2234](#) defines a formal set of rules for a formal
522 unambiguous machine-parsable grammar.

523 The SM ME Addressing Grammar provides unambiguous productions that guarantee that a
524 programmatic parser can be created to detect valid ME addresses. The SM ME Addressing rules
525 provide directives to implementations to guarantee that a UFiT is unique within its immediate
526 container.

527 The following is the SM ME Address Grammar:

```
528 ;;  
529 ;; The SM ME Addressing grammar provides a scheme for validating that  
530 ;; an address presented to a parser is valid according to the rules  
531 ;; specified in the text of this document.  
532 ;;  
533 ;; In the grammar below, the valid Addressing Associations have been  
534 ;; used to formulate valid address patterns by the grammar productions.  
535 ;; The grammar assumes that the implementation has used the  
536 ;; valid Addressing Associations named in tables above to form the  
537 ;; relationships between the instances represented by the UFiT terms in  
538 ;; the address.  
539 ;;
```

```

540 ;; For example, the address /system1/logs1/log2/record1 is a valid
541 ;; address. The grammar specifies that a UFiT term using the UFcT
542 ;; "system" may be followed by a UFiT term using the UFcT "logs", a term
543 ;; using "logs" may be followed by a term using "log", and a term using
544 ;; "log" may be followed by a term using "record". It is up to the
545 ;; implementation to validate that the correct corresponding CIM
546 ;; associations were used in the CIM server to associate the terms in the
547 ;; address.
548
549 ;; Managed Element Address
550 MEAddress = UFiP ; objects
551 MEAddress =/ (UFiP "=>" TargetAssoc [ "=>" UFiP ]) ; associations
552
553 ;; User Friendly instance Paths:
554 ;; /admin1
555 ;; /admin1/<logical paths>
556 ;; /admin1/<physical paths>
557 ;; /system1
558 ;; /hdwrl
559 UFiP = TermSeparator AdminDomainUFcT UFiS
560 UFiP =/ UFiP-Logical
561 UFiP =/ UFiP-Physical
562
563 ;; User Friendly instance Tags:
564 UFiS = %x31-39 *[%x30-39] ; User Friendly instance Suffix
565 UFiT = (UFcT UFiS)
566
567 ;; User Friendly class Tags:
568 ;; All the UFcTs in the Tables plus OEM.
569 UFcT = TableOneUFcT / TableTwoUFcT / TableThreeUFcT / TableFourUFcT
570 UFcT =/ TableFiveUFcT / TableSixUFcT / TableSevenUFcT / TableEightUFcT
571 UFcT =/ TableNineUFcT / TableTenUFcT / TableElevenUFcT
572 UFcT =/ OEMUFcT
573
574 TermSeparator = "\" / "/"
575
576 ;; User-Friendly instance Paths for Logical Containment Hierarchy
577 ;; (Example paths included prior to productions)
578 ;; /admin1/system1/<system components or collections>
579 ;; /admin1/system1/system1/<system components or collections>
580 ;; /admin1/system1/system1/system1/<system components or collections>
581 UFiP-Logical = [ TermSeparator AdminDomainUFcT UFiS ]
582 1*( TermSeparator ComputerSystemUFcT UFiS )
583 [ TermSeparator ( ComponentOfSystem / SystemCollection ) ]
584
585 ;; ComponentOfSystem is a production for immediately-contained elements
586 ;; of a ComputerSystem
587 ComponentOfSystem = Service / SAP / StoragePool / Device / Account
588 ComponentOfSystem =/ ResourceDomain / Software
589

```

```

590 ;; User and Account management
591 ;; /admin1/system1/account2
592 ;; /admin1/system1/account2/identity1
593 Account = AccountUFcT UFiS [TermSeparator IdentityUFcT UFiS]
594
595
596 ;; All ComputerSystem Devices except Sensors, EthernetPort and Processor
597 ;; /admin1/system1/disk3
598
599 Device          = LogicalDeviceUFcT UFiS
600 Device          =/ DeviceTrayUFcT UFiS TermSeparator LogicalDeviceUFcT
601 UFiS
602 Device          =/ SensorUFcT UFiS
603 Device          =/ EnetStack
604 Device          =/ Processor
605
606
607 ;; Service Access Points
608 ;; /admin1/system1/dhpendpt1/remotesap1
609 ;; /admin1/system1/dnsendpt1/remotesap1
610 ;; /admin1/system1/ipendpt1/remotesap1
611
612
613 ;; /admin1/system1/remotesap1
614 SAP             =/ CLPEndptUFcT UFiS
615 SAP             =/ ProtoEndptUFcT UFiS
616 SAP             =/ RemoteSAPUFcT UFiS
617 SAP             =/ SCSIProtoEndptUFcT UFiS
618 SAP             =/ SSHEndptUFcT UFiS
619 SAP             =/ SWidResUFcT UFiS
620 SAP             =/ TCPEndptUFcT UFiS
621 SAP             =/ TelnetEndptUFcT UFiS
622 SAP             =/ TextRedirectSAPUFcT UFiS
623 SAP             =/ IPStack
624 SAP             =/ DNSClientStack
625 SAP             =/ DHCPClientStack
626
627 ;; System services
628 ;; /admin1/system1/actsvcl
629 ;; /admin1/system1/actsvcl/identy2
630
631 Service         = AcctSvcUFcT UFiS
632                 [( TermSeparator IdentityUFcT UFiS )]
633                 [(TermSeparator UserUFcT UFiS ) ]
634
635 ;; /admin1/system1/bootsvc1
636 Service         =/ BootSvcUFcT UFiS
637 Service         =/ NetPortCfgSvcUFcT UFiS
638 Service         =/ PowerManagementSvcUFcT UFiS
639
640 ;; /admin1/system1/clpsvc1/clpendpt2
641 ;; /admin1/system1/sshsvc1/sshendpt1
642 ;; /admin1/system1/telnetvc1/telnetendpt1

```

```

643
644 Service          =/ ProtoSvcUFcT UFiS
645                 [( TermSeparator ProtoEndptUFcT UFiS )
646                 /( TermSeparator CLPEndptUFcT UFiS )
647                 /( TermSeparator SSHEndptUFcT UFiS )
648                 /( TermSeparator TelnetEndptUFcT UFiS ) ]
649                 [TermSeparator TCPEndptUFcT UFiS ) ]
650
651
652 ;; /admin1/system1/rolesvc1
653 ;; /admin1/system1/rolesvc1/privilege1
654
655 Service          =/ RoleSvcUFcT UFiS
656                 [ TermSeparator PrivilegeUFcT UFiS ]
657
658 Service          =/ SharedDeviceMgmtSvcUFcT UFiS
659 Service          =/ SoftwareInstallationSvcUFcT UFiS
660 Service          =/ StorageCfgSvcUFcT UFiS
661 Service          =/ TextRedirectSvcUFcT UFiS
662 Service          =/ TimeSvcUFcT UFiS
663 Service          =/ IpCfgSvcUFcT UFiS
664
665 ;; System storage pool
666 ;; /admin1/system1/storagepool2
667 StoragePool      = StoragePoolUFcT UFiS
668
669 ;; System Admin Domains
670 ;; /admin1/system1
671 AdminDomain      = AdminDomainUFcT UFiS
672 ;; /system1/admin1 /system1/pwrdom1 /system1/coolingdom1
673 ResourceDomain   = (AdminDomainUFcT / ResourceDomainUFcT) UFiS
674
675
676 ;;
677 ;; SystemCollection is a production for named collections that
678 ;; are dedicated to a ComputerSystem
679 ;;
680 SystemCollection = JobQueue / Logs
681 SystemCollection =/ AvailableSW / Sensors / Consoles
682 SystemCollection =/ Group / Capabilities / Capacities
683 SystemCollection =/ Settings / RedundancySet / Role
684
685 ;; Dedicated collection for Capabilities
686 ;; /system1/capabilities1
687 ;; /system1/capabilities1/swinstallsvccap1
688 Capabilities     = CapabilitiesUFcT UFiS
689                 [ ( TermSeparator SharingCapUFcT UFiS )
690                 / ( TermSeparator PowerMgmtCapUFcT UFiS )
691                 / ( TermSeparator SoftwareInstallationSvcCapUFcT UFiS)
692                 / ( TermSeparator StorageCapUFcT UFiS )
693                 / ( TermSeparator StorageCfgCapUFcT UFiS )
694                 / ( TermSeparator DHCPCapUFcT UFiS
695                 / ( TermSeparator BootSvcCapUFcT UFiS ) ]

```

```

696         / ( TermSeparator CLPCapUFcT UFiS )
697         / ( TermSeparator PhyAssetCapUFcT UFiS )
698         / ( TermSeparator TnetCapUFcT UFiS )
699         / ( TermSeparator SSHCapUFcT UFiS )
700         / ( TermSeparator AcctMgtCapUFcT UFiS )
701         / ( TermSeparator RoleMgtCapUFcT UFiS )
702         / ( TermSeparator ProcCapUFcT UFiS )
703         / ( TermSeparator EnabledLogEleCapUFcT UFiS ) ]
704
705 ;; Dedicated collection for Capacities
706 ;; /system1/capacities1
707 ;; /system1/capacities1/configcapacity23
708 Capacities      = CapacitiesUFcT UFiS
709                 [ ( TermSeparator ConfigCapacityUFcT UFiS ) ]
710
711 ;; Dedicated collection for system Consoles
712 ;; /system1/consoles1
713 ;; /system1/consoles1/textredirectsap1
714 Consoles        = ConsolesUFcT UFiS
715                 [ ( TermSeparator TextRedirectSAPUFcT UFiS ) ]
716
717 ;; Ethernet Stack Production
718 EnetStack = EthernetPortUFcT UFiS TermSeparator IPStack
719
720 ;; Dedicated collection for Group
721 ;; /admin1/system1/group5
722 ;; /admin1/system1/group5/identity2
723 Group          = GroupUFcT UFiS
724                 [ ( TermSeparator IdentityUFcT UFiS ) ]
725
726 Role           = RoleUFcT UFiS
727
728
729 ;; Job queue
730 ;; /system1/jobq1
731 ;; /system1/jobq1/job323
732 JobQueue       = JobQueueUFcT UFiS
733                 [ ( TermSeparator ConcreteJobUFcT UFiS ) ]
734
735 ;; Dedicated collection for Logs
736 ;; /system1/logs1
737 ;; /system1/logs1/log1
738 ;; /system1/logs1/log1/record45
739 Logs           = LogsUFcT UFiS
740                 [ ( TermSeparator RecordLogUFcT UFiS ) ]
741                 [ ( TermSeparator LogEntryUFcT UFiS ) ] ]
742
743 ;; Processor Productions
744 Processor = ProcessorUFcT UFiS [ TermSeparator ProcessorCore
745                                 [ TermSeparator HardwareThread ] ]
746
747 ProcessorCore = ProcessorCoreUFcT UFiS
748 HardwareThread = HardwareThreadUFcT UFiS

```

```

749
750 ;; IP Stack and ProtocolEndpoint productions
751 IPStack      = [LANEndPtUFcT UFiS] [ ( TermSeparator IPEndPtUFcT UFiS
752      [(DNSClientStack / DHCPClientStack))] / IPClientStack ]
753
754 DHCPClientStack = TermSeparator DhcpEndPtUFcT UFiS
755      [( TermSeparator RemoteSAPUFcT UFiS )
756      / (TermSeparator DhcpServerUFcT UFiS )]
757
758 DNSClientStack = TermSeparator DnsEndPtUFcT UFiS
759      [( TermSeparator RemoteSAPUFcT UFiS )
760      / (TermSeparator DnsServerUFcT UFiS)]
761
762 IPClientStack = TermSeparator IPEndPtUFcT UFiS
763      [( TermSeparator RemoteSAPUFcT UFiS )
764      / (TermSeparator GatewayUFcT UFiS )]
765
766
767 ;; Dedicated collection for Redundant Devices
768 ;; /system1/redundancyset1
769 ;; /system1/ redundancyset1/fan3
770 ;; /system1/ redundancyset2/powersup2
771 RedundancySet = RedundancySetUFcT UFiS [ TermSeparator LogicalDevice ]
772
773
774 ;; Dedicated collection for Sensors
775 ;; /system1/sensors1
776 ;; /system1/sensors1/tempensor4
777 Sensors      = SensorsUFcT UFiS
778      [ ( TermSeparator SensorUFcT UFiS ) ]
779
780 ;; Dedicated collection for all Settings
781 ;; /system1/settings1
782 ;; /system1/settings1/displaysetting32
783 Settings     = SettingsUFcT UFiS
784      [ (SettingData
785      / (TermSeparator IPAssignmentSettings)
786      / ( TermSeparator BootConfigSetting ) ) ]
787
788 SettingData  = ( TermSeparator AcctSettingUFcT UFiS )
789      / ( TermSeparator CLPSettingUFcT UFiS )
790      / ( TermSeparator SSHSettingUFcT UFiS )
791      / ( TermSeparator StorageSettingUFcT UFiS )
792      / ( TermSeparator TelnetSettingUFcT UFiS )
793
794 ;; Dedicated collection for Boot Configuration SettingData
795 ;; /system1/settings1/bootcfgsetting1
796 ;; /system1/settings1/bootcfgsetting1/bootsetting1
797 ;; /system1/settings1/bootcfgsetting1/bootsrcsetting1
798 ;; /system1/settings1/bootcfgsetting1/bootsrcsetting1/bootsetting2
799 ;;/system1/settings1/bootcfgsetting1/bootsrcsetting1/bootsetting2/bootcfg
800 setting3
801

```

```

802 BootConfigSetting = BootConfigSettingUFcT UFiS
803     [ ( TermSeparator BootSettingDataUFcT UFiS ) /
804       ( SettingData ) /
805       ( TermSeparator BootSourceSetting ) ]
806 BootSourceSetting = BootSourceSettingUFcT UFiS
807     [ ( TermSeparator BootSettingDataUFcT UFiS ) /
808       ( SettingData ) / ( BootConfigSetting ) ]
809
810 ;; IP Assignment Settings for IP Settings
811 IPAssignmentSettings = IPSettingUFcT UFiS
812     [ ( TermSeparator StaticIPSettingUFcT UFiS )
813       / ( TermSeparator DHCPSettingUFcT )
814       / ( TermSeparator DNSGeneralSettingUFcT )
815       / ( TermSeparator DNSSettingUFcT ) ]
816
817 ;; Dedicated collection for Software
818 ;; /system1/availablesw1
819 ;; /system1/availablesw1/swid23
820 ;; //system1/availablesw1/swid23/swid31
821
822 AvailableSW      = AvailableSWUFcT UFiS [TermSeparator Software]
823
824 Software         = SoftwareIdentityUFcT UFiS
825     *(TermSeparator SoftwareIdentityUFcT UFiS)
826     [TermSeparator SWidResUFcT UFiS]
827
828 ;; User-Friendly instance Paths for Physical Containment Hierarchy
829 ;; /admin1/hdwrl
830 ;; /admin1/hdwrl/<physical package paths>
831 UFiP-Physical = [ TermSeparator AdminDomainUFcT UFiS ]
832     TermSeparator HWConcreteCollectionUFcT UFiS
833     [ TermSeparator PhysicalPackage ]
834
835 ;; PhysicalPackage identifies those elements which the Physical
836 ;; Containment Hierarchy allows
837 ;; to be contained within another PhysicalPackage
838 PhysicalPackage = Rack / Chassis / PhysicalFrame / Package / Card
839
840 ;; /admin1/hdwrl/rack1
841 ;; /admin1/hdwrl/rack1/chassis1/...
842 ;; /admin1/hdwrl/rack1/frame1/...
843 ;; /admin1/hdwrl/rack1/card1/...
844 ;; /admin1/hdwrl/rack1/pkg1/...
845 ;; /admin1/hdwrl/rack1/<connector or component>
846 Rack = RackUFcT UFiS
847 Rack =/ RackUFcT UFiS TermSeparator PhysicalPackage
848 Rack =/ RackUFcT UFiS TermSeparator ConnectorSlot
849 Rack =/ RackUFcT UFiS TermSeparator Component
850

```

```

851 ;; /admin1/hdwrl/chassis1
852 ;; /admin1/hdwrl/chassis1/rack1/...
853 ;; /admin1/hdwrl/chassis1/frame1/...
854 ;; /admin1/hdwrl/chassis1/card1/...
855 ;; /admin1/hdwrl/chassis1/pkg1/...
856 ;; /admin1/hdwrl/chassis1/<connector or component>
857 Chassis = ChassisUFcT UFiS
858 Chassis =/ ChassisUFcT UFiS TermSeparator PhysicalPackage
859 Chassis =/ ChassisUFcT UFiS TermSeparator ConnectorSlot
860 Chassis =/ ChassisUFcT UFiS TermSeparator Component
861
862 ;; /admin1/hdwrl/frame1
863 ;; /admin1/hdwrl/frame1/card1/...
864 ;; /admin1/hdwrl/frame1/pkg1/...
865 ;; /admin1/hdwrl/frame1/<connector or component>
866 PhysicalFrame = PhysicalFrameUFcT UFiS
867 PhysicalFrame =/ PhysicalFrameUFcT UFiS TermSeparator PhysicalPackage
868 PhysicalFrame =/ PhysicalFrameUFcT UFiS TermSeparator ConnectorSlot
869 PhysicalFrame =/ PhysicalFrameUFcT UFiS TermSeparator Component
870
871 ;; /admin1/hdwrl/pkg1
872 ;; /admin1/hdwrl/pkg1/card1/...
873 ;; /admin1/hdwrl/pkg1/pkg1/...
874 ;; /admin1/hdwrl/pkg1/<connector or component>
875 Package = PackageUFcT UFiS
876 Package =/ PackageUFcT UFiS TermSeparator PhysicalPackage
877 Package =/ PackageUFcT UFiS TermSeparator ConnectorSlot
878 Package =/ PackageUFcT UFiS TermSeparator Component
879
880 ;; /admin1/hdwrl/card1
881 ;; /admin1/hdwrl/card1/card1/...
882 ;; /admin1/hdwrl/card1/<connector or component>
883 Card = CardUFcT UFiS
884 Card =/ CardUFcT UFiS TermSeparator PhysicalPackage
885 Card =/ CardUFcT UFiS TermSeparator ConnectorSlot
886 Card =/ CardUFcT UFiS TermSeparator Component
887
888 ;; /admin1/hdwrl/.../connector1
889 ;; /admin1/hdwrl/.../slot1
890 ;; /admin1/hdwrl/.../slot1/card1
891 ;; /admin1/hdwrl/.../connector1/pkg1
892 ConnectorSlot = Connector / Slot
893 Connector = ConnectorUFcT UFiS
894 Connector =/ ConnectorUFcT UFiS TermSeparator PhysicalPackage
895 Connector =/ ConnectorUFcT UFiS TermSeparator ConnectorUFcT UFiS
896 Slot = SlotUFcT UFiS
897 Slot =/ SlotUFcT UFiS TermSeparator PhysicalPackage
898 Slot =/ SlotUFcT UFiS TermSeparator SlotUFcT UFiS
899
900
901 ;; /admin1/hdwrl/.../component1
902 ;; /admin1/hdwrl/.../chip1
903 Component = ( ComponentUFcT / ChipUFcT / PhysicalMemoryUFcT ) UFiS

```

```

904
905 ;; The following sections define the terminal productions for the
906 ;; SM ME Addressing Grammar.
907
908 ;; User Friendly class Tags in the Logical Containment Hierarchy
909 ;; in the order introduced in the UFcT tables in this document.
910 ;; The ordering is based upon the UFcT class' end-most subclass.
911
912 ;; special opaque OEM specific UFcT
913 OEMUFcT = "OEM" 1*VCHAR      ; VCHAR: visible characters
914
915 ;; Table 1 CIM_ManagedElement UFcTs
916 TableOneUFcT = PrivilegeUFcT / ConfigCapacityUFcT
917 TableOneUFcT =/ LogEntryUFcT /
918 TableOneUFcT =/ UserUFcT / IdentityUFcT
919
920 PrivilegeUFcT = "privilege"
921 ConfigCapacityUFcT = "configcapacity"
922 LogEntryUFcT = "record"
923 UserUFcT = "user"
924 IdentityUFcT = "identity"
925
926 ;; Table 2 CIM_Capabilities UFcTs
927 TableTwoUFcT = PowerMgmtCapUFcT / SharingCapUFcT
928 TableTwoUFcT =/ SoftwareInstSvcCapUFcT / StorageCapUFcT
929 TableTwoUFcT =/ StorageCfgCapUFcT/ DHCPCapUFcT
930 TableTwoUFcT =/ BootSvcCapUFcT / CLPCapUFcT / PhyAssetCapUFcT
931 TableTwoUFcT =/ TnetCapUFcT / SSHCapUFcT / AcctMgmtCapUFcT
932 TableTwoUFcT =/ RoleMgtCapUFcT / EnabledLogEleCapUFcT
933 TableTwoUFcT =/ ProcCapUFcT
934
935 PowerMgmtCapUFcT = "pwrmgtpcap"
936 SharingCapUFcT = "sharingcap"
937 SoftwareInstSvcCapUFcT = "swinstallsvccap"
938 StorageCapUFcT = "storagecap"
939 StorageCfgCapUFcT = "storagecfgcap"
940 DHCPCapUFcT = "dhccap"
941 BootSvcCapUFcT = "bootsvccap"
942 CLPCapUFcT = "clpcap"
943 PhyAssetCapUFcT = "phyassetcap"
944 TnetCapUFcT = "tnetcap"
945 SSHCapUFcT = "sshcap"
946 AcctMgtCapUFcT = "acctmgtpcap"
947 RoleMgtCapUFcT = "rolemgtpcap"
948 EnabledLogEleCapUFcT = "elecap"
949 ProcCapUFcT = "cpucap"
950
951

```

```

952 ;; Table 3 CIM_SettingData UFcTs
953 TableThreeUFcT = BootConfigSettingUFcT / BootSourceSettingUFcT
954 TableThreeUFcT =/ BootSettingDataUFcT / DisplaySettingUFcT
955 TableThreeUFcT =/ StorageSettingUFcT
956 TableThreeUFcT =/ IPSettingUFcT / StaticIPSettingUFcT / DHCPSettingUFcT
957 TableThreeUFcT =/ DNSSettingUFcT / DNSGeneralSettingUFcT
958 TableThreeUFcT =/ AcctSettingUFcT / CLPSettingUFcT
959 TableThreeUFcT =/ SSHSettingUFcT / TelnetSettingUFcT
960
961 AcctSettingUFcT = "acctsetting"
962 BootConfigSettingUFcT = "bootcfgsetting"
963 BootSourceSettingUFcT = "bootsrcsetting"
964 BootSettingDataUFcT = "bootsetting"
965 CLPSettingUFcT = "clpsetting"
966 DisplaySettingUFcT = "displaysetting"
967 StorageSettingUFcT = "storagesetting"
968 IPSettingUFcT = "ipsetting"
969 SSHSettingUFcT = "sshsetting"
970 StaticIPSettingUFcT = "staticipsetting"
971 TelnetSettingUFcT = "telnetsetting"
972 DHCPSettingUFcT = "dhcpsetting"
973 DNSSettingUFcT = "dnssetting"
974
975 ;; Table 4 CIM_Collection UFcTs
976 TableFourUFcT = CapabilitiesUFcT / CapacitiesUFcT / ConsolesUFcT
977 TableFourUFcT =/ ConcreteCollectionUFcT / GroupUFcT / LogsUFcT
978 TableFourUFcT =/ HWConcreteCollectionUFcT
979 TableFourUFcT =/ RedundancySetUFcT
980 TableFourUFcT =/ SettingsUFcT / SensorsUFcT / AvailableSWUFcT
981 TableFourUFcT =/ RoleUFcT
982
983 ;;
984 ;; The UFiP grammar will enforce the UFcT(s) that can be
985 ;; contained in each collection.
986 ;;
987 CapabilitiesUFcT = "capabilities"
988 CapacitiesUFcT = "capacities"
989 ConsolesUFcT = "consoles"
990 ConcreteCollectionUFcT = "concretecollection"
991 GroupUFcT = "group"
992 LogsUFcT = "logs"
993 HWConcreteCollectionUFcT = "hdwr"
994 RedundancySetUFcT = "redundancysset"
995 SettingsUFcT = "settings"
996 SensorsUFcT = "sensors"
997 AvailableSWUFcT = "availablesw"
998 RoleUFcT = "role"
999

```

```

1000 ;; Table 5 CIM_LogicalElement UFcTs
1001 TableFiveUFcT = SoftwareIdentityUFcT / StoragePoolUFcT
1002
1003 SoftwareIdentityUFcT = "swid" / "swbundle"
1004 StoragePoolUFcT = "storagepool"
1005
1006 ;; Table 6 CIM_EnabledLogicalElement UfctTs
1007 TableSixUFcT = ConcreteJobUFcT / JobQueueUFcT
1008 TableSixUFcT =/ RecordLogUFcT / AccountUFcT
1009 TableSixUFcT =/ ProcessorCoreUFcT / HardwareThreadUFcT
1010
1011 ConcreteJobUFcT = "job"
1012 JobQueueUFcT = "jobq"
1013 RecordLogUFcT = "log"
1014 AccountUFcT = "account"
1015 HardwareThreadUFcT = "hwthread"
1016 ProcessorCoreUFcT = "cpucore"
1017
1018 ;; Table 7 CIM_System UFcTs
1019 TableSevenUFcT = AdminDomainUFcT / ComputerSystemUFcT
1020 TableSevenUFcT =/ ResourceDomainUFcT
1021
1022 AdminDomainUFcT = "admin"
1023 ResourceDomainUFcT = "pwrdom" / "coolingdom"
1024 ComputerSystemUFcT = "system"
1025 ComputerSystemUFcT =/ "modular"
1026 ComputerSystemUFcT =/ "accessserver" / "blockserver"
1027 ComputerSystemUFcT =/ "bridge" / "chassismgr" / "extender"
1028 ComputerSystemUFcT =/ "fileserver" / "firewall" / "gateway" / "hub"
1029 ComputerSystemUFcT =/ "ioserver" / "iphone" / "map" / "management"
1030 ComputerSystemUFcT =/ "medialib" / "mobile" / "modular" / "nas"
1031 ComputerSystemUFcT =/ "nashead" / "printserver" "repeater"
1032 ComputerSystemUFcT =/ "router" / "sp" / "storage" / "storagevlizer"
1033 ComputerSystemUFcT =/ "switch" / "webcache" / "ups"
1034
1035 ;; Table 8 CIM_LogicalDevice UFcTs
1036 TableEightUFcT = BatteryUFcT / CDROMDriveUFcT
1037 TableEightUFcT =/ CoolingDeviceUFcT / DAPortUFcT / DiskDriveUFcT
1038 TableEightUFcT =/ DisketteDriveUFcT / DiskPartitionUFcT / DisplayUFcT
1039 TableEightUFcT =/ DVDDriveUFcT / EthernetPortUFcT / FanUFcT / FCPortUFcT
1040 TableEightUFcT =/ HeatPipeUFcT / IBPortUFcT / KeyboardUFcT
1041 TableEightUFcT =/ LogicalDiskUFcT / LogicalModularUFcT / LogicalPortUFcT
1042 TableEightUFcT =/ MediaAccessUFcT / MemoryUFcT / ModemUFcT
1043 TableEightUFcT =/ NetworkPortUFcT / PCIDeviceUFcT / PCIBridgeUFcT
1044 TableEightUFcT =/ PointingDeviceUFcT / PortControllerUFcT
1045 TableEightUFcT =/ PowerSupplyUFcT / PrinterUFcT / ProcessorUFcT
1046 TableEightUFcT =/ RefrigerationUFcT / SCSIProtoControlUFcT / SensorUFcT
1047 TableEightUFcT =/ SPIPort / StorageVolumeUFcT / StorageExtentUFcT
1048 TableEightUFcT =/ SerialPortUFcT / TapeDriveUFcT / USBPortUFcT
1049 TableEightUFcT =/ WatchdogUFcT / WirelessPortUFcT
1050 TableEightUFcT =/ PortCtrlUFcT / PassThroughModuleUFcT
1051

```

```

1052 BatteryUFcT = "battery"
1053 CDROMDriveUFcT = "cd"
1054 CoolingDeviceUFcT = "cooling"
1055 DAPortUFcT = "daport"
1056 DiskDriveUFcT = "diskdrive"
1057 DisketteDriveUFcT = "floppy"
1058 DiskPartitionUFcT = "diskpartition"
1059 DisplayUFcT = "display"
1060 DVDDriveUFcT = "dvd"
1061 EthernetPortUFcT = "enetport"
1062 FanUFcT = "fan"
1063 FCPortUFcT = "fcport"
1064 HeatPipeUFcT = "heatpipe"
1065 IBPortUFcT = "ibport"
1066 KeyboardUFcT = "keyboard"
1067 LogicalDiskUFcT = "disk"
1068 LogicalModularUFcT = "logicalmodule" / "blademodule" / "linecard"
1069 LogicalModularUFcT =/ DeviceTrayUFcT
1070 DeviceTrayUFcT = "devicetray"
1071 LogicalPortUFcT = "logicalport"
1072 MediaAccessUFcT = "mediaaccess"
1073 MemoryUFcT = "memory"
1074 ModemUFcT = "modem"
1075 NetworkPortUFcT = "netport"
1076 PassThroughModuleUFcT = "passthru"
1077 PCIDeviceUFcT = "pcidev"
1078 PCIBridgeUFcT = "pcibridge"
1079 PointingDeviceUFcT = "pointer" / "mouse" / "trackball"
1080 PointingDeviceUFcT =/ "touchpad" / "touchscreen"
1081 PortControllerUFcT = "portctrl" / "nic" / "hca" / "tca" / "hba"
1082 PowerSupplyUFcT = "pwrsupply"
1083 PrinterUFcT = "printer"
1084 ProcessorUFcT = "cpu"
1085 RefrigerationUFcT = "refrigeration"
1086 SCSIProtoControlUFcT = "scsiprotctrl"
1087 SensorUFcT = "sensor" / "currentsensor" / "tachsensor"
1088 SensorUFcT =/ "tempsensor" / "voltsensor" / "humiditysensor"
1089 SensorUFcT =/ "countersensor" / "swsensor" / "locksensor"
1090 SensorUFcT =/ "smokesensor" / "airsensor" / "presencesensor"
1091 SensorUFcT =/ "numsensor"
1092 SPIPort = "spiport"
1093 StorageVolumeUFcT = "storagevol"
1094 StorageExtentUFcT = "storageext"
1095 SerialPortUFcT = "serialport"
1096 TapeDriveUFcT = "tapedrive"
1097 USBPortUFcT = "usbport"
1098 WatchdogUFcT = "watchdog"
1099 WirelessPortUFcT = "wifiport"
1100 PortCtrlUFcT = "portctrl"
1101
1102 ;; Convenience Production for all LogicalDevice instances
1103 ;; immediately contained in a System
1104 LogicalDeviceUFcT = TableEightUFcT

```

```

1105
1106 ;; Table 9
1107 TableNineUFcT = AcctSvcUFcT / BootSvcUFcT
1108 TableNineUFcT =/ NetPortCfgSvcUFcT
1109 TableNineUFcT =/ PowerManagementSvcUFcT / ProtoSvcUFcT / RoleSvcUFcT
1110 TableNineUFcT =/ SharedDeviceMgmtSvcUFcT / SoftwareInstallationSvc
1111 TableNineUFcT =/ StorageCfgSvcUFcT
1112 TableNineUFcT =/ TextRedirectSvcUFcT / TimeSvcUFcT
1113 TableNineUFcT =/ IpCfgSvcUFcT
1114
1115 AcctSvcUFcT = "acctsvc"
1116 BootSvcUFcT = "bootsvc"
1117 NetPortCfgSvcUFcT = "netportcfgsvc"
1118 PowerManagementSvcUFcT = "pwrmgtsvc"
1119 ProtoSvcUFcT = "protosvc" / "clpsvc" / "telnetsvc" / "sshsvc"
1120 RoleSvcUFcT = "rolesvc"
1121 SharedDeviceMgmtSvcUFcT = "shareddevicesvc"
1122 SoftwareInstallationSvc = "swinstallsvc"
1123 StorageCfgSvcUFcT = "storagecfgsvc"
1124 TextRedirectSvcUFcT = "textredirectsvc"
1125 TimeSvcUFcT = "timesvc"
1126 IPCfgSvcUFcT = "ipcfgsvc"
1127
1128 ;; Table 10
1129 TableTenUFcT = CLPEndptUFcT / ProtoEndptUFcT / RemoteSAPUFcT
1130 TableTenUFcT =/ SCSIProtoEndptUFcT / SSHEndptUFcT / SWidResUFcT
1131 TableTenUFcT =/ TCPEndptUFcT / TelnetEndptUFcT / TextRedirectSAPUFcT
1132 TableTenUFcT =/ IPEndPtUFcT / DNSEndPtUFcT / LANEndPtUFcT
1133 TableTenUFcT =/ DhcpEndPtUFcT / DnsServerUFcT / DhcpServerUFcT
1134 TableTenUFcT =/ GatewayUFcT
1135
1136 CLPEndptUFcT = "clpendpt"
1137 ProtoEndptUFcT = "protoendpt"
1138 RemoteSAPUFcT = "remotesap"
1139 SCSIProtoEndptUFcT = "scsiendpt"
1140 SSHEndptUFcT = "sshendpt"
1141 SWidResUFcT = "swidres"
1142 TCPEndptUFcT = "tcpendpt"
1143 TelnetEndptUFcT "telnetendpt"
1144 TextRedirectSAPUFcT = "textredirectsap"
1145 IPEndPtUFcT = "ipendpt"
1146 DNSEndPtUFcT = "dnsendpt"
1147 LANEndPtUFcT = "lanendpt"
1148 DhcpEndPtUFcT = "dhcpendpt"
1149 DnsServerUFcT = "dnsserver"
1150 DhcpServerUFcT = "dhcpserver"
1151 GatewayUFcT = "gateway"
1152
1153
1154 ;; User Friendly class Tags in the Physical Containment Hierarchy:
1155

```

```

1156 ;; Table 11 CIM_PhysicalElement UFcTs
1157 TableElevenUFcT = CardUFcT / ChassisUFcT / ChipUFcT / ComponentUFcT
1158 TableElevenUFcT =/ ConnectorUFcT / FrameUFcT / PackageUFcT
1159 TableElevenUFcT =/ PhysicalMemoryUFcT / RackUFcT / SlotUFcT
1160
1161 CardUFcT = "card"
1162
1163 ChassisUFcT = "chassis"
1164 ChassisUFcT =/ "laptop" / "desktop" / "tower"
1165 ChassisUFcT =/ "storagechassis" / "notebook" / "mainchassis"
1166 ChassisUFcT =/ "expansion" / "peripheralchassis" / "subchassis"
1167
1168 ChipUFcT = "bga" / "chip" / "dimm" / "dip" / "fpbga" / "lc"
1169 ChipUFcT =/ "lga" / "plcc" / "pga" / "propchip" / "qfp" / "rimm"
1170 ChipUFcT =/ "sip" / "simm" / "smd" / "sodimm" / "soj" / "ssmp"
1171 ChipUFcT =/ "soic" / "srimm" / "tqfp" / "tsop" / "zip"
1172
1173 ComponentUFcT = "component"
1174
1175 ConnectorUFcT = "connector"
1176
1177 PhysicalFrameUFcT = "frame"
1178
1179 ;; generic package UFcT
1180 PackageUFcT = "pkg"
1181 PackageUFcT =/ "backplanepkg" / "batterypkg" / "bladepkg" / "bladexpkg"
1182 PackageUFcT =/ "cardpkg" / "chassispkg" / "cpupkg" / "diskpkg"
1183 PackageUFcT =/ "fanpkg" / "framepkg" / "memorypkg" / "modulepkg"
1184 PackageUFcT =/ "pwrpkg" / "pwrsrcpkg" / "rackpkg" / "sensorpkg"
1185 PackageUFcT =/ "storagepkg"
1186
1187 PhysicalMemoryUFcT = "bram" / "cache" / "cdram" / "ddr" / "dram" / "edo"
1188 PhysicalMemoryUFcT =/ "edram" / "eeprom" / "eprom" / "flash" / "pmem"
1189 PhysicalMemoryUFcT =/ "ram" / "rdram" / "sdram" / "sgram" / "sram"
1190 PhysicalMemoryUFcT =/ "synchdram" / "vram"
1191
1192 RackUFcT = "rack"
1193
1194 SlotUFcT = "slot"
1195
1196 ;;
1197 ;; Associations as the address targets:
1198 TargetAssoc = "BindsToLANEndpoint" / "ConcreteDependency"
1199 TargetAssoc =/ "UseOfLog" / "ElementSettingData"
1200 TargetAssoc =/ "IsSpare" / "LogicalIdentity"
1201 TargetAssoc =/ "AssignedIdentity" / "AssociatedCacheMemory"
1202 TargetAssoc =/ "HostedCollection" / "OwningJobElement"
1203 TargetAssoc =/ "RoleLimitedToTarget" / "ServiceAccessBySAP"
1204 TargetAssoc =/ "SettingsDefineCapabilities" / "SystemPackaging"
1205 TargetAssoc =/ "SystemComponent" / "SystemDevice"
1206 TargetAssoc =/ "HostedService" / "HostedAccessPoint"
1207 TargetAssoc =/ "OwningCollectionElement" / "HostedStoragePool"
1208 TargetAssoc =/ "ConcreteComponent" / "MemberOfCollection"

```

1209 TargetAssoc =/ "OrderedMemberOfCollection" / "LogManagesRecord"
1210 TargetAssoc =/ "HostedJobDestination" / "JobDestinationJobs"
1211 TargetAssoc =/ "MemberOfCollection"
1212 TargetAssoc =/ "Container" / "LogicalIdentity"
1213 TargetAssoc =/ "PackageInConnector"
1214 TargetAssoc =/ "AssociatedCooling" /"ElementSoftwareIdentity"
1215 TargetAssoc =/ "ServiceAvailableToElement"
1216 TargetAssoc =/ "SAPAvailableForElement"
1217 TargetAssoc =/ "DeviceSAPImplementation"
1218 TargetAssoc =/ "ElementCapabilities"
1219 TargetAssoc =/ "AllocatedFromStoragePool" / "Realizes"
1220 TargetAssoc =/ "ControlledBy" / "ComputerSystemPackage"
1221 TargetAssoc =/ "InstalledSoftwareIdentity" / "HostedDependency"
1222 TargetAssoc =/ "SuppliesPower" / "AssociatedSensor"
1223 TargetAssoc =/ "SCSIInitiatorTargetLogicalUnitPath"/ "BindsTo"
1224 TargetAssoc =/ "ProvidesEndpoint" / "SAPSAPDependency"
1225 TargetAssoc =/ "ServiceServiceDependency"
1226 TargetAssoc =/ "AccountOnSystem"
1227 TargetAssoc =/ "AssociatedPowerManagementService"
1228 TargetAssoc =/ "SharingDependency" / "ElementCapacity"
1229 TargetAssoc =/ "ServiceAffectsElement"
1230 TargetAssoc =/ "BindsTo"
1231 TargetAssoc =/ "RemoteAccessAvailableToElement"
1232 TargetAssoc =/ "OrderedComponent"
1233 TargetAssoc =/ "ServiceAvailableToElement"
1234

1235 **6 Summary**

1236 Several key components combine to form the technical foundation that enables SM ME Addressing
1237 to provide a simple user-friendly unique address path to a CIM instance. These key components are:

- 1238 • SM Profiles
- 1239 • SM ME Grammar
- 1240 • CIM Model

1241 The following sections describe the role that each of these key components plays in the SM Address
1242 scheme.

1243 **6.1 SM Profiles**

1244 CIM is a very rich modeling environment. It provides many ways to model computing environments,
1245 allowing different vendors to choose different model components to model their computing
1246 equipment and environments. While this open-ended modeling behavior is desirable in the
1247 theoretical sense, it poses grave difficulties in a practical environment. For example, the open-ended
1248 modeling often creates a situation where vendors' tools cannot interoperate because they were built
1249 using different notions of the computing model.

1250 The SM Containment Hierarchy defined in section 4 forms the basis for SM ME Addressing by
1251 identifying the hierarchical containment associations between objects. *SMASH Implementation*
1252 *Requirements* defines the model components that implementations must support. The SM ME
1253 Addressing rules define how implementations create UFiTs for instances of classes specified in the
1254 SM profiles.

1255 If implementations follow the models specified in the profiles and the addressing rules, then fully
1256 qualified UFiPs are guaranteed to exist for all managed instances in the implementations'
1257 namespace. The SM profiles guarantee that all conforming implementations provide the same MEs
1258 and supporting associations. The SM Containment Hierarchy guarantees that these MEs are
1259 addressable.

1260 **6.2 SM ME Grammar**

1261 The SM ME Addressing Grammar (see section 5.3) provides unambiguous productions that
1262 guarantee that a programmatic parser can be coded to detect valid ME addresses. Furthermore, it
1263 guarantees that each valid production will eventually terminate in a unique UFiT or UFcT. The
1264 difference between the UFcT and the UFiT is the unique instance suffix, which identifies a particular
1265 CIM instance of a class.

1266 **6.3 CIM Model**

1267 The CIM schema defines classes and associations between instances of CIM classes. The SM ME
1268 Addressing Specification provides the rules for subsets of CIM associations called Addressing
1269 Associations, which are used to form ME addresses. These rules are used to validate the terms in a
1270 UFiP. The UFiTs on each side of the association path must have the proper object type and role.
1271 CIM defines the properties for each object class and its subclasses.

1272

Appendix A – Considerations for Implementation

1273
1274
1275
1276
1277
1278
1279

Table 15 shows several examples of valid SM UFiTs that are based on the instance's UFcT and are unique within a container. The first column shows the container instance's UFcT above the corresponding CIM class name in parentheses. Similarly, the second column shows the contained instance's UFcT with its corresponding CIM class name in parentheses. The third column shows the contained instance's UFiT above one or more examples of a complete UFiP. Note that the difference between the UFcT and the UFiT is the instance suffix (a non-negative integer). The instance suffix must be unique within the scope of the container class.

1280

Table 15 UFiT Examples

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 /admin1/hdwr1/bladepkg2/pwrpkg2 /admin1/hdwr1/bladepkg5/pwrpkg2 /admin1/hdwr1/rack7/bladechassis2/bladepkg8/pwrpkg3
pkg (CIM_PhysicalPackage)	chip (CIM_Chip)	chip1 /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
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 /admin1/hdwr1/bladepkg1/slot1

Container UFcT (CIM Class Name)	Contained UFcT (CIM Class Name)	Contained Instance UfiT Complete UfiP
card (CIM_Card)	slot (CIM_Slot)	slot1 /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/system2/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 (CIM_ConcreteCollection)	clpcap log clpsetting (CIM_ManagedElement)	capabilities1, logs1, clpsetting1 /admin1/system1/capabilities1/clpcap1 /admin1/system1/logs1/log1 /admin1/system1/settings1/clpsetting1

1281