Date: February 2008

Application Management and System Monitoring for CMS Systems

Beta 2 with change bars

OMG Document Number: dtc/2008-02-02

USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

The material in this document details an Object Management Group specification in accordance with the terms, conditions and notices set forth below. This document does not represent a commitment to implement any portion of this specification in any company's products. The information contained in this document is subject to change without notice.

LICENSES

The companies listed above have granted to the Object Management Group, Inc. (OMG) a nonexclusive, royalty-free, paid up, worldwide license to copy and distribute this document and to modify this document and distribute copies of the modified version. Each of the copyright holders listed above has agreed that no person shall be deemed to have infringed the copyright in the included material of any such copyright holder by reason of having used the specification set forth herein or having conformed any computer software to the specification.

Subject to all of the terms and conditions below, the owners of the copyright in this specification hereby grant you a fully-paid up, non-exclusive, nontransferable, perpetual, worldwide license (without the right to sublicense), to use this specification to create and distribute software and special purpose specifications that are based upon this specification, and to use, copy, and distribute this specification as provided under the Copyright Act; provided that: (1) both the copyright notice identified above and this permission notice appear on any copies of this specification; (2) the use of the specifications is for informational purposes and will not be copied or posted on any network computer or broadcast in any media and will not be otherwise resold or transferred for commercial purposes; and (3) no modifications are made to this specification. This limited permission automatically terminates without notice if you breach any of these terms or conditions. Upon termination, you will destroy immediately any copies of the specifications in your possession or control.

PATENTS

The attention of adopters is directed to the possibility that compliance with or adoption of OMG specifications may require use of an invention covered by patent rights. OMG shall not be responsible for identifying patents for which a license may be required by any OMG specification, or for conducting legal inquiries into the legal validity or scope of those patents that are brought to its attention. OMG specifications are prospective and advisory only. Prospective users are responsible for protecting themselves against liability for infringement of patents.

GENERAL USE RESTRICTIONS

Any unauthorized use of this specification may violate copyright laws, trademark laws, and communications regulations and statutes. This document contains information which is protected by copyright. All Rights Reserved. No part of this work covered by copyright herein may be reproduced or used in any form or by any means--graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems--without permission of the copyright owner.

DISCLAIMER OF WARRANTY

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE OBJECT MANAGEMENT GROUP AND THE COMPANIES LISTED ABOVE MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE.
IN NO EVENT SHALL THE OBJECT MANAGEMENT GROUP OR ANY OF THE COMPANIES LISTED ABOVE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The entire risk as to the quality and performance of software developed using this specification is borne by you. This disclaimer of warranty constitutes an essential part of the license granted to you to use this specification.

RESTRICTED RIGHTS LEGEND

Use, duplication or disclosure by the U.S. Government is subject to the restrictions set forth in subparagraph (c) (1) (ii) of The Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013 or in subparagraph (c)(1) and (2) of the Commercial Computer Software - Restricted Rights clauses at 48 C.F.R. 52.227-19 or as specified in 48 C.F.R. 227-7202-2 of the DoD F.A.R. Supplement and its successors, or as specified in 48 C.F.R. 12.212 of the Federal Acquisition Regulations and its successors, as applicable. The specification copyright owners are as indicated above and may be contacted through the Object Management Group, 140 Kendrick Street, Needham, MA 02494, U.S.A.

TRADEMARKS

MDA®, Model Driven Architecture®, UML®, UML Cube logo®, OMG Logo®, CORBA® and XMI® are registered trademarks of the Object Management Group, Inc., and Object Management Group™, OMG™ , Unified Modeling Language™, Model Driven Architecture Logo™, Model Driven Architecture Diagram™, CORBA logos™, XMI Logo™, CWM™, CWM Logo™, IIOP™ , MOF™ , OMG Interface Definition Language (IDL)™ , and OMG SysML™ are trademarks of the Object Management Group. All other products or company names mentioned are used for identification purposes only, and may be trademarks of their respective owners.

COMPLIANCE

The copyright holders listed above acknowledge that the Object Management Group (acting itself or through its designees) is and shall at all times be the sole entity that may authorize developers, suppliers and sellers of computer software to use certification marks, trademarks or other special designations to indicate compliance with these materials.

Software developed under the terms of this license may claim compliance or conformance with this specification if and only if the software compliance is of a nature fully matching the applicable compliance points as stated in the specification. Software developed only partially matching the applicable compliance points may claim only that the software was based on this specification, but may not claim compliance or conformance with this specification. In the event that testing suites are implemented or approved by Object Management Group, Inc., software developed using this specification may claim compliance or conformance with the specification only if the software satisfactorily completes the testing suites.

OMG's Issue Reporting Procedure

All OMG specifications are subject to continuous review and improvement. As part of this process we encourage readers to report any ambiguities, inconsistencies, or inaccuracies they may find by completing the Issue Reporting Form listed on the main web page *http://www.omg.org*, under Documents, Report a Bug/Issue (*http://www.omg.org/technology/agreement.htm*).

Table of Contents

Preface

About the Object Management Group

OMG

Founded in 1989, the Object Management Group, Inc. (OMG) is an open membership, not-for-profit computer industry standards consortium that produces and maintains computer industry specifications for interoperable, portable, and reusable enterprise applications in distributed, heterogeneous environments. Membership includes Information Technology vendors, end users, government agencies, and academia.

OMG member companies write, adopt, and maintain its specifications following a mature, open process. OMG's specifications implement the Model Driven Architecture® (MDA®), maximizing ROI through a full-lifecycle approach to enterprise integration that covers multiple operating systems, programming languages, middleware and networking infrastructures, and software development environments. OMG's specifications include: UML® (Unified Modeling Language™); CORBA® (Common Object Request Broker Architecture); CWM™ (Common Warehouse Metamodel); and industry-specific standards for dozens of vertical markets.

More information on the OMG is available at *http://www.omg.org/*.

OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. A Specifications Catalog is available from the OMG website at:

http://www.omg.org/technology/documents/spec_catalog.htm

Specifications within the Catalog are organized by the following categories:

OMG Modeling Specifications

- UML
- MOF
- XMI
- CWM
- Profile specifications.

OMG Middleware Specifications

- CORBA/IIOP
- IDL/Language Mappings
- Specialized CORBA specifications
- CORBA Component Model (CCM).

Platform Specific Model and Interface Specifications

- CORBAservices
- CORBAfacilities
- OMG Domain specifications
- OMG Embedded Intelligence specifications
- OMG Security specifications.

All of OMG's formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. at:

OMG Headquarters
140 Kendrick Street
Building A, Suite 300
Needham, MA 02494
USA
Tel: +1-781-444-0404
Fax: +1-781-444-0320
Email: *pubs@omg.org*

Certain OMG specifications are also available as ISO standards. Please consult *http://www.iso.org*

Typographical Conventions

The type styles shown below are used in this document to distinguish programming statements from ordinary English. However, these conventions are not used in tables or section headings where no distinction is necessary.

Times/Times New Roman - 10 pt.: Standard body text

Helvetica/Arial - 10 pt. Bold: OMG Interface Definition Language (OMG IDL) and syntax elements.

`Courier - 10 pt. Bold:` Programming language elements.

Helvetica/Arial - 10 pt: Exceptions

Note – Terms that appear in *italics* are defined in the glossary. Italic text also represents the name of a document, specification, or other publication.

Issues

The reader is encouraged to report any technical or editing issues/problems with this specification to *http://www.omg.org/technology/agreement.htm*.

1 Scope

The term C4I (Command, Control, Communications, Computers, Intelligence) covers – in a very general definition – an information system with associated (or integrated) sensors and actuator subsystems.

Within the wide domain of C4I systems, we distinguish the domain of naval Combat Management Systems. C4I systems are located on a naval ship, that assist the command team in its responsibility for execution of a mission. The Naval CMS systems' main capabilities encompass awareness of situation around the ship (or a group of ships within a naval force or theater) using sensors, exchange and fusion of this information into a tactical picture to support recognition of threats and response to those threats using actuators such as torpedo, missile and gun systems. Other capabilities of a Naval CMS include those frequently called Command Support capabilities, which in general are concerned with preparation of the ship's mission. They also include the preparation and supervision of execution of diverse plans, as well as reception and interpretation of communication from external parties (other vessels or shore-based parties).

The problem of application management and system monitoring in CMS systems can be characterized as follows:

- CMS systems aboard naval vessels are built upon a huge variety of underlying computing platforms (including both hardware and software infrastructures). Those differences are manifest within the scope of a single CMS on a single vessel as well as within a family of CMS systems in a product-line organization of a CMS provider. In a single vessel case, applications may vary from hard-real-time applications used in fire control, to non-real-time office-like applications used in Command Support activities. Within a product line, CMS providers want to achieve long-term and large-scale reuse of applications on changing computing platforms. Both phenomena require a consistent and if possible unified form of application management.
- These computing platforms manage applications in different, often proprietary and incompatible ways. There is a lack of formalized consensus on basic terms such as “an application”, “a software system” and such. Few of the available relevant standards only address parts of the problem area.
- The different available platforms (e.g., process based and component based frameworks) are not consistent in even simple forms of application management such as starting and stopping.
- Specific aspects of application management such as quality-of-service (e.g., end-to-end time constraints on starting and stopping of applications) which are relevant to the naval CMS case are in general not addressed.
- There is a lack of overarching application management framework that would enable application builders and CMS system integrators to abstract from platform dependencies.

Similar problems can be identified in other military and non-military systems, especially in the area of crisis management systems. Therefore, the solutions, when developed, are expected to be reusable in other domains (dual-use).

2 Conformance Criteria

The conformance criteria of an implementation w.r.t this specification is stated through the concepts of "profiles" and "PSM."

The *Normal profile* is formalized by:

- The following packages (see Table 2.1 for further information)
- LW Logging
- AMSM Management without the AMS_HWFilter and AMS_RTHWIndication classes

- Supported Application Model
- Application
- Application Specification
- Application Deployment
- Application Deployment Specification
- And the following classes and their associations of the "Logical Hardware" package: AMS_ComputerSystem, AMS_OperatingSystem, AMS_Host.

Other known profiles are:

- *Maximum Control profile*: this profile adds the optional methods AMS_ExecutableSoftwareElement::Load, the optional state “LOADED, the optional Transitions “LOAD,” “START,” “UNLOAD,” and “LOAD_DIRTY,” the optional items AMS_LOAD, AMS_START, AMS_UNLOAD, AMS_LOAD_DIRTY, and AMS_LOADED.

This profile is intended for operating systems that cannot load an executable onto memory without effectively executing it such as Linux, Windows systems, and so on.

- *Fault Tolerance Management profile*: This profile adds the optional classes AMS_RedundancyEltState, AMS_RedundancyGroup, AMS_ReplicationStyle, AMS_RedundancyGroupManagement, and the optional associations RedInitState, CIM_SystemComponent (between AMS_Application and AMS_Redundancy-Group), CIM_RedundancyComponent (between AMS_ExecutableSoftwareElement and AMS_RedundancyGroup), AMS_RG, AMS_RedundancyFeature, and Status.
- *Load Balancing Management profile*: This profile adds the optional classes AMS_RedundancyEltState, AMS_LoadBalancingGroup, AMS_BalancingStyle, AMS_LoadBalancingManagement, and the optional associations RedInitState, CIM_SystemComponent (between AMS_Application and AMS_LoadBalancing-Group), CIM_RedundancyComponent (between AMS_ExecutableSoftwareElement and AMS_Load-BalancingGroup), AMS_LB, AMS_LoadBalancinfFeature, and Status.
- *Hardware System Management profile*: This profile adds the "Logical Hardware Specification" package and the classes and associations of the "Logical Hardware" package that are not present in the *Normal profile*.

Known PSMs are:

- *IDL PSM*: see Chapter 8 “OMG CORBA/IDL Platform Specific Model”
- *XML PSM*: see Chapter 9 “XML Platform Specific Model”
- *CIM PSM*: see Chapter 10 “DMTF CIM Managed Object Format (MOF) Platform Specific Model”
- *DCPS/f PSM*: see Chapter 11 “DDS/DCPS Platform Specific Model”
- *DCPS/m PSM*: see Chapter 11 “DDS/DCPS Platform Specific Model”
- *HPI PSM*: this PSM is defined by the "XML for HPI" PSM (see Chapter 12 “XML for HPI Platform Specific Model”)

This PSM has the same level of information as the matching part of the PIM. It must be understood as a model that deals with the necessary information to figure out the cross-referencing of the AMSM model with the HPI model, thus allowing the use of the HPI API within an AMSM service implementation.

Obviously, this PSM cannot be implemented when the "*Hardware System Management profile*" is not present.

The *"IDL PSM," "CIM PSM,"* and *"DCPS/f PSM"* are known as the *"Core PSMs."*

Subsequently, the following rules applies to a compliant implementation (the key words "MUST" and "MAY" in this paragraph are to be interpreted as described in RFC 2119):

- The *"Normal profile*" MUST be implemented (i.e., the software part is mandatory).
- The other profiles MAY be implemented (i.e, Hardware, Load Balancing, Fault Tolerance, and maximum control capabilities are optional but compatible).
- The *"XML PSM"* MUST be implemented (i.e, XML is mandatory).
- At least one of the *"Core PSMs"* MUST be implemented (i.e., the AMSM service is always accessible through IDL, CIM, or DCPS interfaces).
- When one of the *"Core PSMs"* is implemented, the other PSM belonging to the *"Core PSMs"* set MAY be implemented (i.e., IDL, CIM, and DCPS/f PSM are compatible).
- The *"HPI PSM"* MAY be implemented if and only if the "*Hardware System Management profile*" is also implemented (i.e., the use of HPI is optional, dependent of the hardware profile and compatible with other profiles).
- The *"DCPS/m PSM"* MAY be implemented when the *"DCPS/f PSM"* is not implemented (i.e., it is possible – but not required – because there may exist valid reasons in particular circumstances to ignore the implementation of DDS/ DCPS).

A summary of the aforementioned rules is provided in the following tables:

Table 2.1 Profile vs. Packages

Profile vs. Packages rules[a]		**Profiles**				
		Normal	**HW System Management**	**Fault Tolerance**	**Load Balancing**	**Maximum Control**
Packages	LW Logging	F	F	F	F	F
	AMSM Management	P (no HW, LB and FT elements)	+ HW elements	+ FT elements	+ LB elements	P
	Supported Application Model	F	F	F	F	F
	Application	P (no LB and FT elements)	P	+ FT elements	+ LB elements	+ opt. methods
	Application Spec	P (no LB and FT elements)	P	+ FT elements	+ LB elements	P
	Application Deployment	F	F	F	F	F
	Application Deployment Spec.	F	F	F	F	F
	Logical Hardware	P (classes for hosts)	F	P	P	P
	Logical Hardware Spec.	/	F	/	/	/

a. Table legend:
A At least one among them must be implemented
P Partial
F Full

Table 2.2 Profiles vs. Implementation

Profiles vs. Implementation rules		Profiles				
		Normal	**HW System Management**	**Fault Tolerance**	**Load Balancing**	**Maximum Control**
Implementation	Must	X				
	May		X	X	X	X

Table 2.3 Implementation vs. PSMs

Implementation vs. PSMs rules			Implementation	
			Must	**May**
PSMs		XML	X	
	Core PSMs	IDL	At least one among them	
		CIM		
		DCPS/f		
		DCPS/m		X
		HPI		X

3 Normative References

The following normative documents contain provisions, which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply.

- [**CIM**] CIM Schema, DMTF, V2.9.0. 5 Jan, 2005.
- [**CIMMOF**] CIM 2.2 Specification, DMTF, 14 June 1999. (DSP0004).
- [**CIMHTTP**] CIM Operations over HTTP, DMTF, V1.1.6 January, 2003. (DSP0200).
- [**CIMXML**] Representation of CIM in XML, DMTF, V2.1.2 May, 2002. (DSP0201).
- [**CORBA**] Common Object Request Broker Architecture: Core Specification, OMG, V3.0.3 (formal/04-03-12).

- [**D&C**] Deployment and Configuration of Component-based Distributed Applications, OMG, V4.0 (formal/06-04-02).
- [**DDS**] Data Distribution Service for Real-time Systems Specification, OMG, V1.0, (formal/04-12-02).
- [**HPI**] Hardware Platform Interface Specification, SA Forum, HPI-B-01.01 January, 2005.
- [**JEE5**] Java Enterprise Edition v5.0, Sun Microsystems.

Issue 11516 - Lightweight logging specification is a formal spec.

- [**LWLOG**] Lightweight Logging Service, OMG (formal/05-02-02).
- [**REGEXP**] Chapter 9, “Regular Expressions” of the *Base Definitions volume* of IEEE Std 1003.1-2001. http://www.opengroup.org/onlinepubs/009695399/basedefs/ xbd_chap09.html.
- [**SNMP**] Simple Network Management Protocol.
- [**WBEM**] The Web Based Enterprise Management Initiative. See the latest version available at http://www.dmtf.org/standards/wbem/.
- [**XMLSchema**] XML Schema,W3C Recommendation, 28 October 2004. Latest version at http://www.w3.org/TR/xmlschema-1/ and http://www.w3.org/TR/xml-schema-2/.

4 Terms and Definitions

4.1 General Definitions

Architecture Board (AB) - The OMG plenary that is responsible for ensuring the technical merit and MDA-compliance of RFPs and their submissions.

Board of Directors (BoD) - The OMG body that is responsible for adopting technology.

Common Object Request Broker Architecture (CORBA) - An OMG distributed computing platform specification that is independent of implementation languages.

Common Warehouse Metamodel (CWM) - An OMG specification for data repository integration.

CORBA Component Model (CCM) - An OMG specification for an implementation language independent distributed component model.

Interface Definition Language (IDL) - An OMG and ISO standard language for specifying interfaces and associated data structures.

Mapping - Specification of a mechanism for transforming the elements of a model conforming to a particular metamodel into elements of another model that conforms to another (possibly the same) metamodel.

Metadata - Data that represents models. For example, a UML model; a CORBA object model expressed in IDL; and a relational database schema expressed using CWM.

Metamodel - A model of models.

Meta Object Facility (MOF) - An OMG standard, closely related to UML, that enables metadata management and language definition.

Model - A formal specification of the function, structure and/or behavior of an application or system.

Model Driven Architecture (MDA) - An approach to IT system specification that separates the specification of functionality from the specification of the implementation of that functionality on a specific technology platform.

Normative – Provisions that one must conform to in order to claim compliance with the standard. (as opposed to non-normative or informative which is explanatory material that is included in order to assist in understanding the standard and does not contain any provisions that must be conformed to in order to claim compliance).

Normative Reference – References that contain provisions that one must conform to in order to claim compliance with the standard that contains said normative reference.

Platform - A set of subsystems/technologies that provide a coherent set of functionality through interfaces and specified usage patterns that any subsystem that depends on the platform can use without concern for the details of how the functionality provided by the platform is implemented.

Platform Independent Model (PIM) - A model of a subsystem that contains no information specific to the platform, or the technology that is used to realize it.

Platform Specific Model (PSM) - A model of a subsystem that includes information about the specific technology that is used in the realization of it on a specific platform, and hence possibly contains elements that are specific to the platform.

Request for Proposal (RFP) - A document requesting OMG members to submit proposals to the OMG's Technology Committee. Such proposals must be received by a certain deadline and are evaluated by the issuing task force.

Task Force (TF) - The OMG Technology Committee subgroup responsible for issuing an RFP and evaluating submission(s).

Technology Committee (TC) - The body responsible for recommending technologies for adoption to the BoD. There are two TCs in OMG – *Platform TC* (PTC), that focuses on IT and modeling infrastructure related standards; and *Domain TC* (DTC), that focus on domain specific standards.

Unified Modeling Language (UML) - An OMG standard language for specifying the structure and behavior of systems. The standard defines an abstract syntax and a graphical concrete syntax.

UML Profile - A standardized set of extensions and constraints that tailors UML to particular use.

XML Metadata Interchange (XMI) - An OMG standard that facilitates interchange of models via XML documents.

4.2 Definitions Specific to this Specification

Application - The set of one or more Executable Software Elements that perform a defined set of functionality. An Application is a structural construct that defines the set of elements that need to be executed to provide the desired functionality. Starting (and stopping) an Application results in configuring and starting (or stopping) the defined set of Executable Software Elements.

Application Model - The platform required by the application (e.g. process model and container model) or supported by part of the system topology.

Application Specification - The generalized configuration information for the set of Executable Software Elements that make up the Application. This information includes configuration settings common across software elements.

Configuration - The set of deployment information such as parameters, rules, and hardware resources needed to support control and monitoring of an application.

Deployment Configuration - A defined set of Software Systems, Applications, and/or Executable Software Elements with explicit parameterization and host mapping for each of the Executable Software Elements (including the executable software elements that comprise the Applications and Software Systems). This defines a static software-to-hardware mapping. A Deployment Configuration may map to an Executable Software Element, an Application, a Software System, or may contain a combination of Executable Software Elements, Applications, and/or Software Systems. Note: future RFPs will define Deployment Configurationsto support dynamic configurations that allow the specific set of Executable Software Elements, parameterization, and host mappings to be determined at run-time.

Executable Software Element - An executable program or equivalent (script, .jar file, single executable CORBA component, etc.) that is be directly managed (i.e., started, stopped, configured, and monitored) by the application management service. This does not include elements such as libraries, which are not executed by themselves.

Executable Software Element Specification - The generalized configuration information for an Executable Software Element, including hardware and OS requirements, information on how to start and stop the software element (e.g., how and where to load the software element, the model [process, container and type of container] the element conforms to, how to configure environment variables and arguments, pre-configuration commands and checks, post-configuration commands and checks).

Host - A computing element of the system topology on which an Executable Software Element can run.

Node - A processing unit such as a server, a workstation or a tightly coupled collection of processor boards within e.g., a VME rack (VME is a flexible open-ended bus system which makes use of the Eurocard standard).

Service Infrastructure - A set of run-time Executable Software Elements and support files that provide the application management service functionality.

Software System - A set of one or more Applications or Software Systems that operate together to provide a higher level set of functionality. A Software System is a structural construct that defines a hierarchical view of the Software Systems and Applications that need to be run to provide the desired higher level functionality. Starting (and stopping) a Software System results in configuring and starting (or stopping) the set of Executable Software Elements defined in each of the Applications under the top-level Software System or under any level of the included Software Systems.

Software System Specification - The generalized configuration information for the set of Software Systems and Applications that make up the Software System. This information includes configuration settings common across all software systems, applications, and software elements.

System Infrastructure - The set of elements such as computers, processors, communication networks. The infrastructure includes both hardware as well as software.

System Topology - The set of computing, storage, and network components on which the application management services and the Executable Software Elements being managed will run.

System Topology Specifications - Information on the expected system topology including network connectivity, hardware resources, and operating system configuration.

4.3 Acronyms and Abbreviations

AIS	Application Interface Specification A standard for application management, developed by the SA Forum
AMS	The trigram used before the name of the classes and associations specifically designed for this specification (i.e., not get from DMTF/CIM).
AMSM	The name of this specification and of the technical services.
CIM	Common Information Model A standard for system administration developed by DMTF
CMIP	Common Management Information Protocol
CMS	(Naval) Combat Management System
CORBA	Common Object Request Broker Architecture
DCOM	Distributed Component Object Model
DCPS	Data-Centric Publish-Subscribe
DDS	Data Distribution Service
DMI	Desktop Management Interface
DMTF	Distributed Management Task Force (cf. www.dmtf.org)
ESE	Executable Software Element
FRU	Field Replaceable Unit
HPI	Hardware Platform Interface A standard for hardware elements management, developed by the SA Forum
HTTP	HyperText Transfer Protocol
LWLOG	Lightweight Logging Service
MIB	Management Information Base
MOF	Managed Object Format (not to be confused with Meta Object Facility) Textual notation used by DMTF to represent CIM models
OM	*Object Manager*
OMG	Object Management Group (cf. www.omg.org)
OP	*Object Provider*
RFP	Request For Proposal
SA Forum	Service Availability Forum (cf. www.saforum.org)
SMS	Systems Management Specification Future combination of HPI and AIS, to be developed by the SA Forum
SNMP	Simple network Management Protocol
UML	Unified Modeling Language
W3C	World Wide Web Consortium (cf. www.w3c.org)
WBEM	Web-Based Enterprise Management
XML	eXtensible Mark-up Language

5 Symbols

There are no symbols applicable in this specification.

6 Additional Information

6.1 Overall Design Rationale

6.1.1 Design Overview

The design of the specification follows the following principles:

- Maximum use possible of existing standard DMTF CIM. The CIM is a widely accepted standard for management of software and hardware systems. As such, it is deemed to form a good basis for CMS AMSM standardization. The specification selects the subset of CIM relevant for the CMS domain and extends it where needed.
- Inclusion of HPI-based hardware monitoring as optional PSM. This is motivated by the fact that CIM does not model hardware elements to the detail level required by AMSM RFP.
- A set of PSMs covering a variety of platform technologies: CORBA, DDS/DCPS, XML, DMTF CIM Managed Object Format (MOF), HPI.
- A hierarchical 3-level model of software systems, applications and software executable elements.
- A division between design-time and run-time information of software and hardware entities.
- A flexible deployment model allowing user defined conditions and actions to be defined.

The following sections provide an introduction to CIM and HPI which form the basis of the specification.

6.1.2 CIM Overview

This is a PIM based upon a standard which exists in the world of commercial Enterprise Management Systems, the DMTF Common Information Model [**CIM**]. CIM provides a standard set of object oriented models in UML and the Managed Object Format [**MOF**] which cover a large variety of devices and can be extended as required for adding new features on an evolving base of models. The MOF language is based upon IDL.

CIM is a standard within the Web Based Enterprise Management [**WBEM**] framework of the DMTF. The goal of WBEM is to unify management of distributed heterogeneous computing systems. This is accomplished by defining a comprehensive and extensible information model of system components, an encoding of CIM in XML [**CIMXML**], and the transport of CIM via HTTP [**CIMHTTP**].

The CIM schemas are updated on a continuing basis to extend support to new areas and to refine models as experience is gained. The MOF specification defines the basic language and rules of constructing CIM schemas. MOF is revised on a much less frequent basis than the CIM schemas.

Utilizing models based upon CIM provides a large base of existing and proven models for basic enterprise management to build from, as well as a framework for extending and creating new models to meet the requirements. In addition, using CIM/WBEM provides a means of integrating a CMS management system with standard COTS enterprise and network management systems.

CIM models a software or hardware system as a collection of component models connected via associations. A specific instance of a system is modeled as a collection of instances of component models and associations. A specific instance of a model is identified by the set of KEY attributes of the model. The MOF standard defines a URI syntax to identify a specific instance of a modeled component.

Although CIM models can be visualized in UML, the definition of the basic language of CIM schemas is found in the [**MOF**] specification. Few of the details of MOF syntax are required for understanding this specification. However, there are several features of MOF which are somewhat unusual and do effect the specification. One of the important features is a restriction in MOF that a derived class can only define new KEY attributes if the superclass does not already contain KEY attributes.

The large variety of models in the base CIM schemas is both a strength and a weakness of CIM. There is usually more than one way (and frequently several ways) to model a given system in CIM, and deciding on the particular schemas to use can be a challenge. In this specification, we have made choices on which subset of CIM models to use. In many cases, other CIM models could just as well be used to meet the requirements.

All models within this specification are derived from standard CIM schemas. As such, we do not list all inherited attributes and methods of the base CIM schemas as they are detailed in [**CIM**]. We only provide an exhaustive list of attributes and methods for classes which are extensions defined in this specification.

6.1.3 HPI Overview

The SA Forum Hardware Platform Interface (HPI) specifies a generic mechanism to monitor and control highly available systems. The ability to monitor and control these systems is provided through a consistent, platform independent set of programmatic interfaces. The HPI specification provides data structures and functional definitions to interact with manageable subsets of a platform or system.

The HPI allows applications and middleware (“HPI User”) to access and manage hardware components via a standardized interface. Its primary goal is to allow for portability of HPI User code across a variety of hardware platforms.

In essence, the HPI model is comprised of four basic concepts – Sessions, Domains, Resources, and Entities – each of which is described briefly below.

6.1.3.1 HPI Entities

Starting at the basic foundation of the HPI model, entities represent the physical components of the system. Each entity has a unique identifier, called entity path, which is defined by the component’s location in the physical containment hierarchy of the system.

An entity’s manageability is modeled in HPI by management instruments, which are defined in resource data records associated with the entity. These management instruments are the mechanisms by which HPI Users control and receive information about the state of the system.

Entity management via the HPI may include any combination of the following functions:

- Reading values related to the operation or health of a component. This ability is modeled as a “Sensors” associated with the entity.
- Controlling aspects of the operation of a component. This ability is modeled as a “Controls” associated with the entity, plus special functions to control the powering and resetting of a component.

- Reporting inventory and static configuration data. This data is reported as a the “Inventory Data Repository” associated with the entity.
- Operating watchdog timers on components. Watchdog timers may cause implementation-defined actions to occur when the timers expire. The ability to operate watchdog timers is modeled via “Watchdog Timers” associated with the entity.
- Announcing status and fault condition information on a component. This ability is modeled as an “Annunciators” associated with the entity.

6.1.3.2 HPI Resources

Resources provide management access to the entities within the system. Each resource is responsible for managing and presenting to the HPI User the entities that it is managing. Additionally, resources may provide the following functions:

- Monitoring and controlling the insertion and removal of components in the system as it operates. This is reported through the interface as “Hot Swap” events and controlled via a set of “Hot Swap” functions.
- Storing a historical log of resource events for later retrieval. This storage and retrieval mechanism is modeled as a “Resource Event Log” contained in the resource.
- Updating management parameters, storing new parameters in non-volatile storage.

Figure 6.1 shows an example of the relations between HPI resources and entities. In this example:

- The boxes in the middle represent the basic “attributes” and “interfaces” which are called “Resource Data Records” in HPI.
- Blue boxes (with italicized text) represent inventory data (product id, serial number, etc.) which is practically mandatory for any entity.
- Green boxes (with underlined text) in represent sensors.
- Red boxes (with bold text) represent controllers.
- The tree structure on the right represents the entities, with their relative position. This is represented in HPI by the “entity path.”
- The yellow circles on the left represent the resources, and the arrows show which “data record” is associated with each resource.

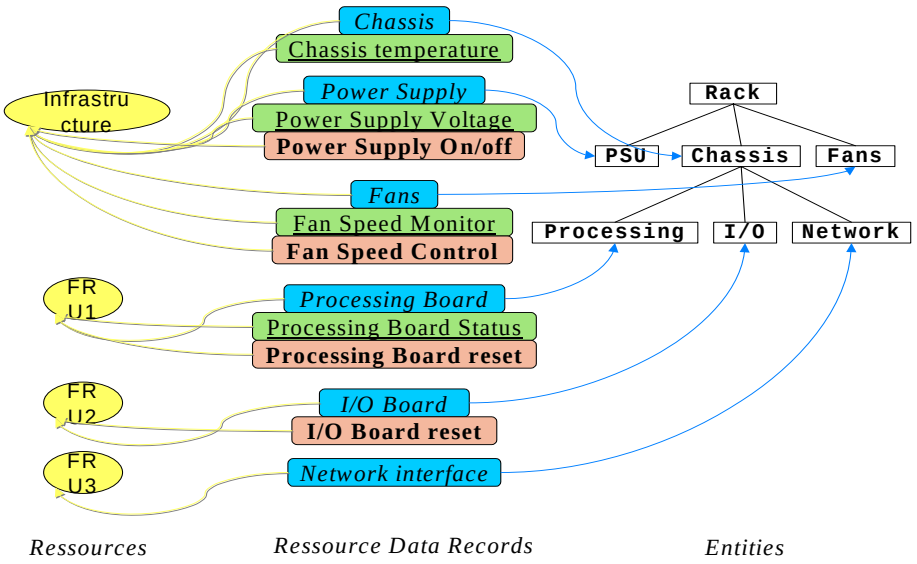


Figure 6.1 - Example of HPI Resources and Entities

6.1.4 HPI Domains

The HPI view of a system is divided into one or more domains, where a domain provides access to some set of the resources within the system. A domain represents some part of the system that is capable of being managed by an HPI User; many systems may have a single domain, whereas systems that have areas dedicated to separate tasks, for example, may manage these through separate domains. Additionally, domains provide the following functions:

- Forwarding resource generated events. Events generated by resources within a domain are disseminated to HPI Users that are subscribed to receive domain events.
- Storing a historical log of events from the resources in the domain for later retrieval. This storage and retrieval mechanism is modeled as a “Domain Event Log” contained in the domain.
- Monitoring and controlling the insertion and removal of components in the system as it operates. This is reported through the interface as “Hot Swap” events, and reflected in a Resource Presence Table (RPT) accessible via the domain.
- Maintaining a table of current fault conditions within the domain.
- Maintaining a table of peer and/or tiered domains associated with the domain.

6.1.4.1 HPI Sessions

Sessions provide access to an HPI implementation. This is accomplished by opening an HPI session within a domain; one HPI User can have multiple sessions open at once, and there can be multiple concurrent open sessions within a domain. It is intended that, in future releases, access control to the HPI will be performed at the session level to allow sessions to have different permission settings. Sessions also provide access to events that originate in the domain for which the session was opened.

6.1.4.2 HPI versus CIM Physical Elements

HPI entities represent the physical components of the system, and correspond to CIM Physical Elements. The information provided by the “entity path” is equivalent to the information provided by CIM “Location” class and “ConntainedLocation” relation.

In CIM, the hierarchical view based on physical assembly is optional, whereas it is mandatory in HPI.

This AMSM standard requests that the hierarchical view be based on physical assembly, to the extent needed to identify and locate the Field Replaceable Units (FRUs) which comprise a computer system.

The definition of an FRU is common to the CIM and HPI: an FRU is a physical element that may be easily removed, added, or replaced while the system is in operation, with minimal impact on the operational capability.

The identification of FRUs which comprise a computer system may be defined statically or discovered dynamically. Both approaches are allowed; the comparison between a “discovered” and an “expected” configuration is not required to be accomplished by the implementation, but the implementation should allow the application software to carry out this comparison (and, for example, an application can decide if a mismatch between the “discovered” and “expected” configuration is to be considered as a computer system fault, or just a restriction on the set of possible configurations.

In operational conditions, for the type of systems in the scope of this AMSM standard, it is not necessarily required to retrieve detailed diagnostics about each FRU. A basic requirement, in case of computer system malfunction, is to identify which FRU is to be replaced.

6.1.4.3 Standardization concern

This AMSM specification is based on standardization efforts within DMTF and the SA-Forum, leveraging consistencies and complementarities between their respective CIM and HPI standards. These are ongoing standardization efforts and are expected to evolve and change after the time of writing this specification – the version of these standards used in this specification are listed in the Normative Reference section. The authors are aware of these non-OMG standardization activities, and are committed to avoiding overlap, and where overlap is needed, to ensure consistency with the OMG AMSM standard.

6.2 Changes or extensions required to adopted OMG specifications

No changes to UML 2.0 or other OMG specifications are required.

6.3 Acknowledgements

- Progeny Systems Corporation
- SELEX Sistemi Integrati (SI)
- THALES
- Themis Computer

7 Platform Independent Model (PIM)

7.1 General View

7.1.1 Packages

In order to break down the overall model in a modular way such that interdependencies and complexity are minimized, two dimensions were considered:

1. Hardware vs. Software vs. Deployment (i.e., Software on Hardware).
2. Run-Time (monitoring) classes vs. Specification Classes.

Based on that reflection, concepts could have been gathered either under three packages: hardware, software, and deployment; or under two packages: run-time and specification. Yet, each of these modeling methods would have been partial and biased.

Hence, this specification keeps at the same root level all the packages implied: run-time hardware (“Logical Hardware”), specification hardware (“Logical Hardware Specification” package), run-time software (“Application” package), specification software (“Application Specification” package), run-time deployment (“Application Deployment” package), and specification deployment (“Application Deployment Specification” package).

Yet these packages can still be spread on the two previous dimensions, as shown in the table:

Package	Hardware	Software	Deployment
Run-Time	Logical Hardware	Application	Application Deployment
Specification	Logical Hardware Specification	Application Specification	Application Deployment Specification

The following graph shows all packages of the AMSM service.

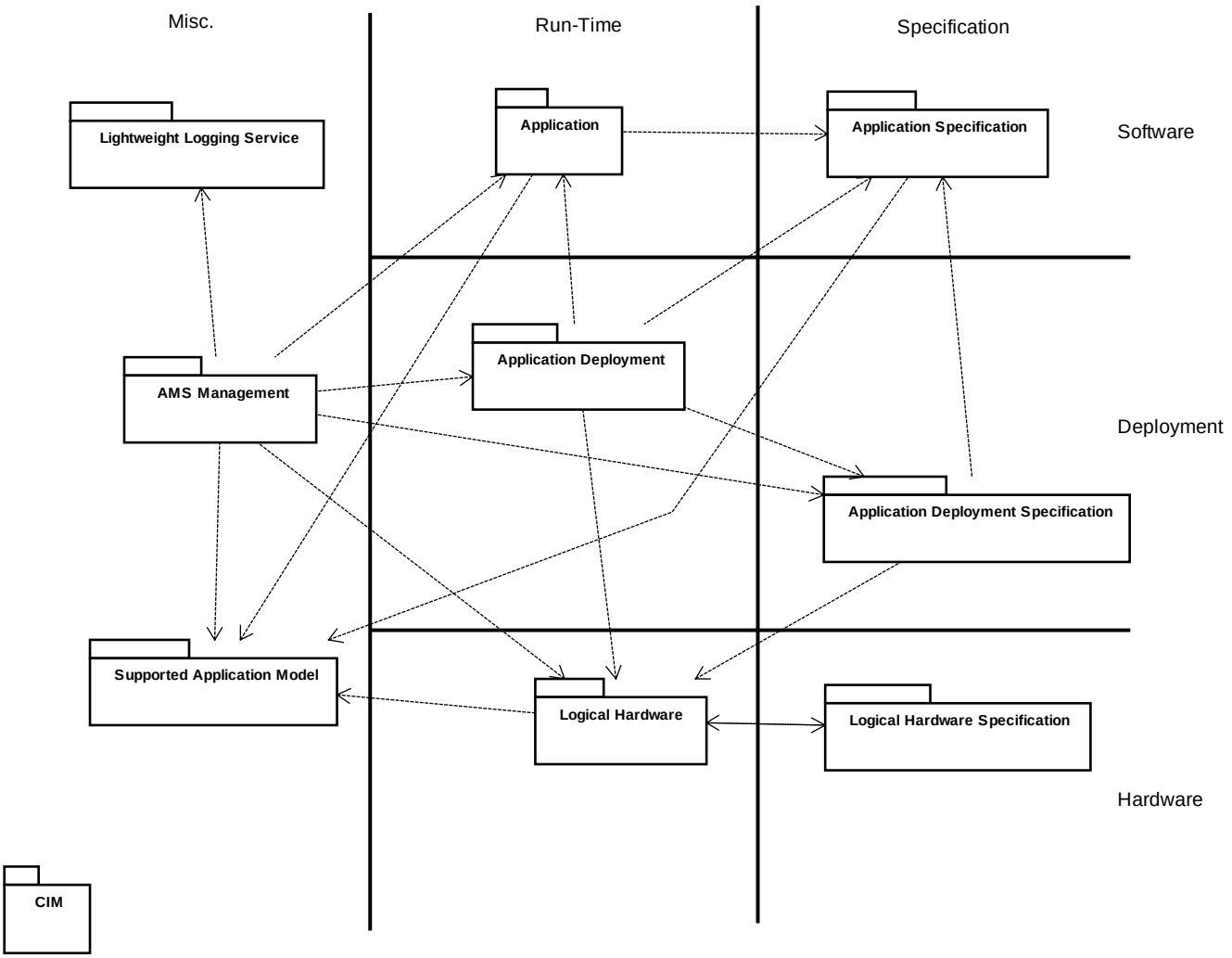


Figure 7.2 - Packages

In the next section, each of these packages will be specified in turn.

7.1.2 Conventions

7.1.2.1 Naming convention

The AMSM service reuses the DMTF CIM naming convention, which is based on key attributes and so-called “weak” associations. In the models shown hereafter, UML constraints have been used to present these naming conventions:

- The constraint {key} on an attribute of a class denotes that this attribute is used to construct the global name of the element.
- The constraint {key} on a side of an aggregation link from the class A to the class B denotes that the global name of an instance of B is constructed by concatenating:

- the global name of the instance of A it is linked with, and
- its global name (given by its {key} constrained attribute).

7.1.2.2 Other constraints

The constraint {override} denotes an attribute or a method inherited from CIM and which multiplicity and/or type have been modified accordingly to the DMTF CIM rules.

The constraints {optional MC}, {optional FTM}, {optional LBM} and {optional HSM} indicate an item that belongs to one of the non-mandatory profiles (cf. Chapter 2): MC for the "Maximum Control profile," FTM for the "Fault Tolerance Management profile," LBM for the "Load Balancing Management profile," and HSM for the "Hardware System Management profile."

Note – For the sake of clarity, associations between optional classes are not noted as optionals but are optionals.

7.1.2.3 Data types

The following table defines the data types used in the specification and that are not classes:

Data type	Definition	Comment
Collection<Type>	A collection of the “Type”	“Type” is a data type or a class. This data type does not preclude any implementation style.
datetime	String of 25 chars of the format yyyymmddhhmmss.mmmmmmsutc where: yyyy is 4 digit year mm is month dd is day hh is 24 hr hour ss is seconds mmmmmm is number of microseconds s is "+" or "-" sign of the offset from utc or ":" if the offset is meaningless,(such as in an interval of time) utc is the offset in minutes	This type is used for all dates.
String	UCS-2 string	ISO/IEC 10646 encoding form: Universal Character Set coded in 2 octets.
uint8	at most 8 bits long unsigned integer	
uint16	at most 16 bits long unsigned integer	
uint32	at most 32 bits long unsigned integer	
uint64	at most 64 bits long unsigned integer	

7.1.2.4 Eluded operations definitions

Some operations need to be typed as collection of known classes. In these cases, the data type “Collection<*Type*>” will be used.

In order to simplify diagrams and to take into account all the specificities of the Platforms which are aimed by PSM, it will be considered that every attribute on a class goes with a mechanism allowing a client to get and set the value of this attribute.

This mechanism could be a couple of get/set methods (CORBA/IDL PSM) or a generic mechanism induced by the PSM itself (DMTF/CIM PSM and DDS/DCPS PSM).

In the same way, all compositions from a class stereotyped as <<enum>> go with a get/set mechanism.

Furthermore, all other associations between classes, which are not stereotyped as <<enum>>, go with a mechanism allowing to get the collection of elements associated with others.

In the rest of this chapter, “CIM” is referring to the DMTF CIM (Common Information Model), “MOF” to the DMTF CIM MOF (Managed Object Format) and “CIM documentation” to [CIM].

7.1.3 AMS Management Package

The "AMS Management" package holds the main entry points of the AMSM service: the AMS_ConfManagement, AMS_HWManagement, AMS_DeploymentConfManagement, AMS_SystemManagement, AMS_ApplicationManagement, AMS_ESEManagement, AMS_LoadBalancingManagement, and AMS_RedundancyGroupManagement classes. These interfaces allow to get status and to subscribe to status changes. This subscription on hardware or software status changes implies callbacks and information on it. These data are gathered in the AMS_*Indication and AMS_*Status classes.

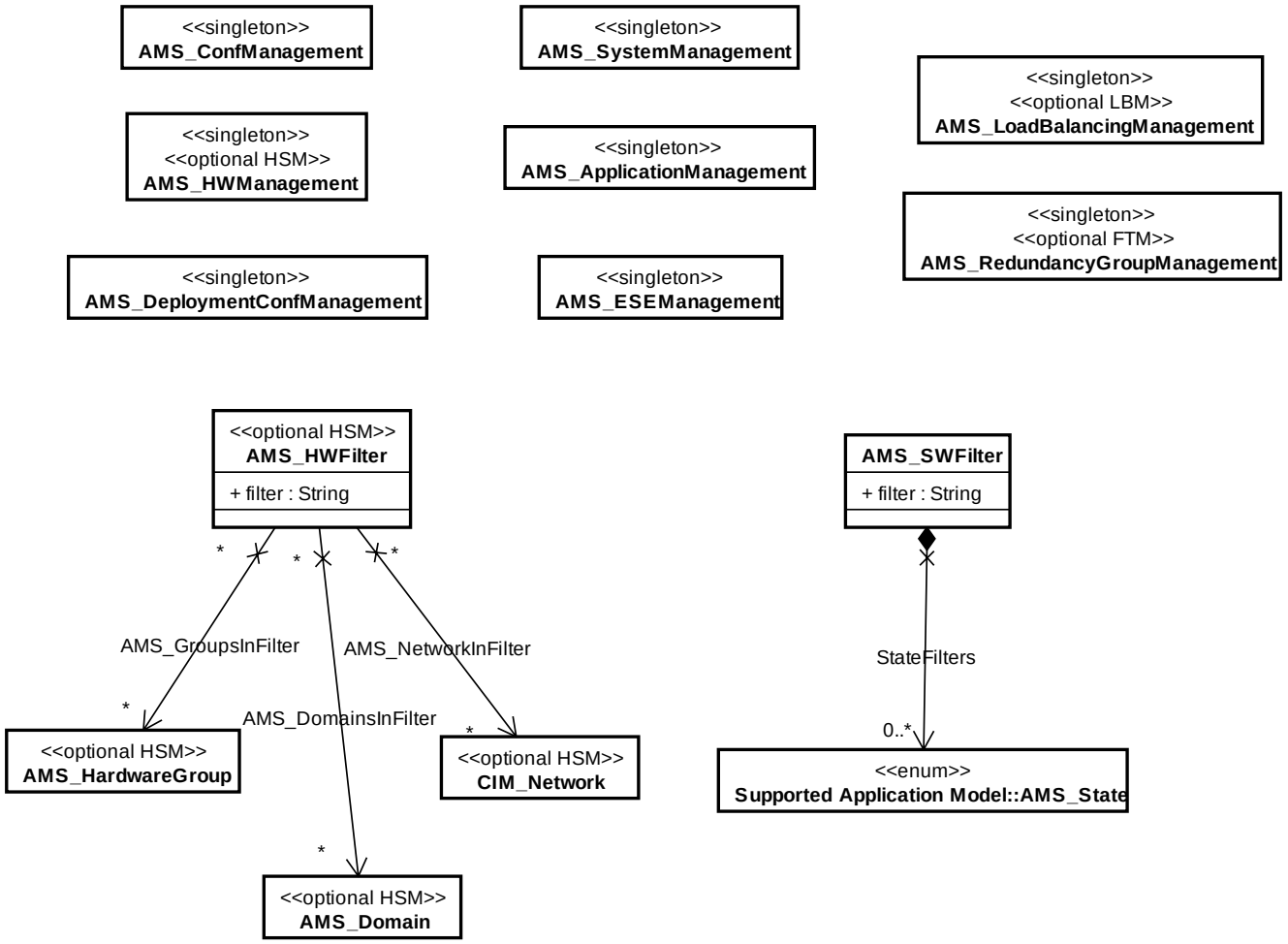


Figure 7.3 - AMS Management class

This diagram displays:

- The 8 classes that provide the entry points of the AMSM service with the operations that give access to the remainder of the model (these operations are not shown in the diagram for clarity):
  - AMS_HWManagement and AMS_PhysicalHWManagement for hardware
  - AMS_DeploymentConfManagement for the deployment configurations
  - AMS_SystemManagement for systems
  - AMS_ApplicationManagement for applications
  - AMS_ESEManagement for executable software Elements
  - AMS_LoadBalancingManagement for load balancing groups
  - AMS_RedundancyGroupManagement for redundancy groups
- The two classes that model filter parameters in some operations of preceding classes:

- AMS_SWFilter for filters on software
- AMS_HWFilter for filters on hardware

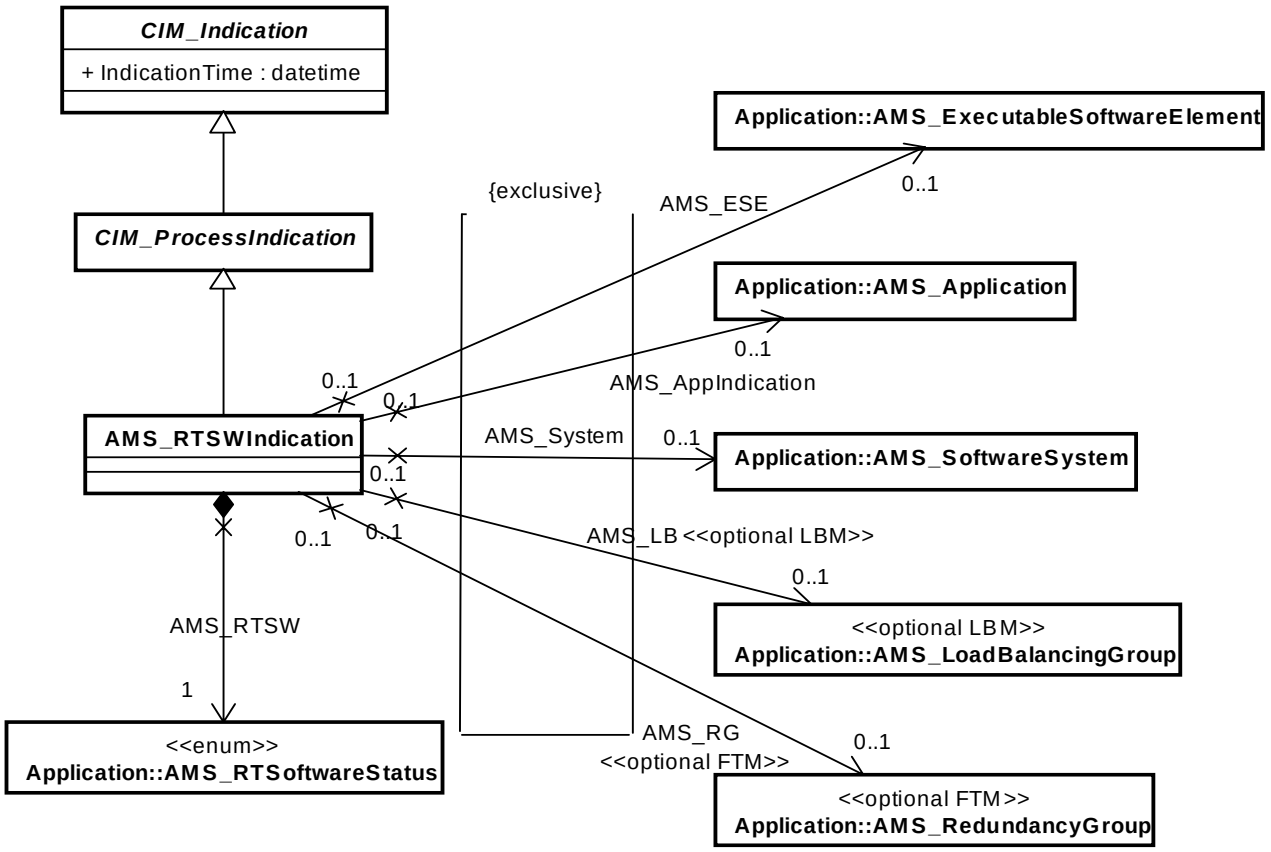


Figure 7.4 - Software Indication Class

In the previous diagram, the AMS_RTSWIndication class gathers the data returned when the state of software changes. Such information deals with the state of an instance of one of the following classes ("exclusive" constraint):

- an executable software element
- an application
- a system
- a load balancing group
- a redundancy group

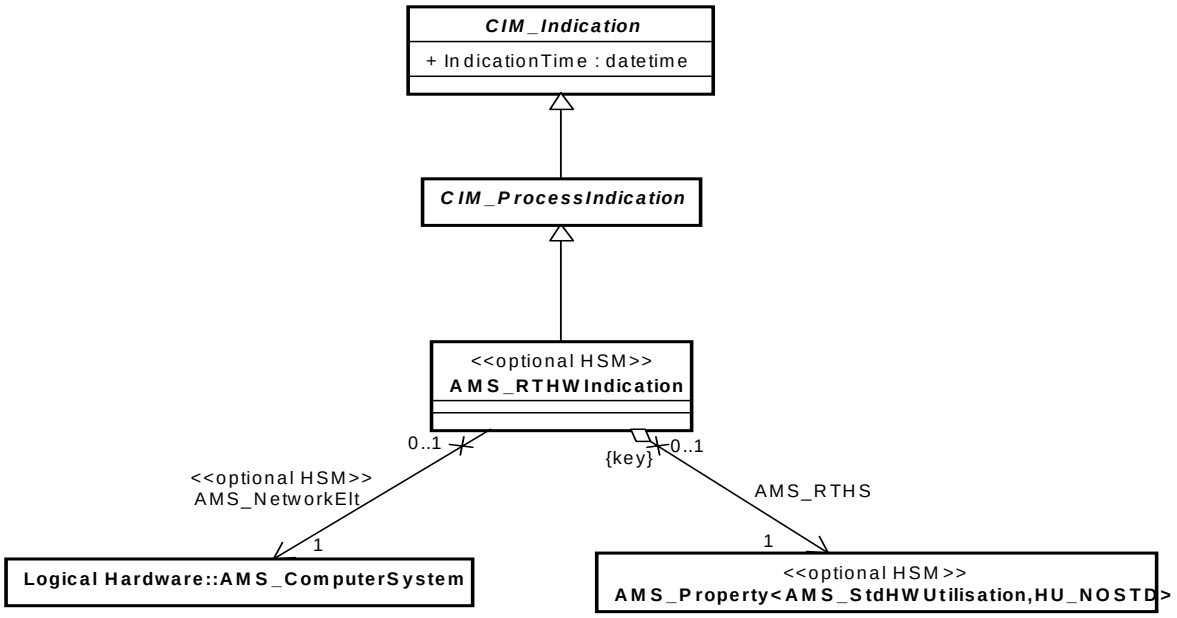


Figure 7.5 - Hardware Indication Class

In this diagram, the AMS_RTHWIndication class gathers the data returned when the state of computers (AMS_ComputerSystem) changes.

Figure 7.6 - Log class

This last diagram displays the log model of the AMSM service. This log model uses fully the interfaces of the lightweight log service (cf. [**LWLOG]**).

7.1.3.1 AMS_ApplicationManagement Class

This class is the interface to access the application information. Its operations allow:

- Subscription to periodic state updates of some applications.
- Subscription to the change of state of some applications.
- An access to other interfaces for a more specific view on executable software elements.

Operations	*Parameters*	*Parameters type*
GetApplication	return type	Collection<AMS_Application>
	filter	AMS_SWFilter
SubscribeApplicationStatusChange	return type	uint16
	filter	AMS_SWFilter
	out subscriptionID	uint32
SubscribeApplicationStatus	return type	uint16
	delay	uint16
	filter	AMS_SWFilter
	out subscriptionID	uint32
Unsubscribe	return type	uint16
	subscriptionID	uint32

The following sections review the operation in detail.

7.1.3.1.1 GetApplication (filter: AMS_SWFilter)

This operation returns all applications that match the specified filter.

7.1.3.1.2 SubscribeApplicationStatus (delay: uint16, filter: AMS_SWFilter, out subscriptionID : uint32)

This operation subscribes to periodic status updates of applications that match the specified "filter."

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The effect of this operation is subscription to AMS_RTSWIndication class.

This operation shall return AMS_BADFILTER if the “filter” parameter is wrong.

7.1.3.1.3 SubscribeApplicationStatusChange (filter: AMS_SWFilter, out subscriptionID : uint32)

This operation subscribes to the status modifications of applications which match the specified "filter."

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The effect of this operation is subscription to AMS_RTSWIndication class.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

7.1.3.1.4 Unsubscribe (subscriptionID: uint32)

This operation removes a previous subscription.

This operation shall return AMS_BADSUBSCRIPTIONID if the parameter is erroneous..

7.1.3.2 AMS_ConfManagement Class

This class is the interface to access the loading and unloading of configuration files.

Operations	*Parameters*	*Parameters type*
LoadConfiguration	return type	uint16
	file	String
UnloadConfiguration	return type	uint16
	file	String
GetLastError	return type	AMS_ErrorStruct

7.1.3.2.1 LoadConfiguration (file: String)

This operation loads a configuration file. This configuration file must conform to the XML PSM (cf. section 2.6).

7.1.3.2.2 UnloadConfiguration (file: String)

This operation unloads a configuration file. This configuration file must have been loaded previously through the LoadConfiguration method.

7.1.3.2.3 GetLastError ()

This operation gives a description of the last error that occurred.

Special cares have to be taken by implementations in order to deal with multi-threading issues.

7.1.3.3 AMS_DeploymentConfManagement Class

This class is the interface to access the deployment configuration information. Its operations allow:

- access to other interfaces for a more specific view on elements.

Operations	*Parameters*	*Parameters type*
GetDeploymentConfiguration	return type	Collection<AMS_DeploymentConfiguration>

The following sections review the operation in detail.

7.1.3.3.1 GetDeploymentConfiguration ( )

This operation returns all deployment configurations.

7.1.3.4 AMS_ErrorStruct

This structure models the error as returned by the AMS_ConfManagement::GetLastError method.

The attributes give:

- A string message
- An error number (the errno of POSIX on POSIX systems) if available (-1 if not)
- An uint16 with the code of the error (cf. Section 7.1.13.1)

Issue 11508 - Reference of an action in AMS_ErrorStruct

- The element that caused the error referenced by a string that contains the full name of this element. When an action fails to be executed, this attributes gives a reference to the CIM_Action that actually fails.

Attributes	*Type*
Message	String
Number	uint16
Code	uint16
Element	String

7.1.3.5 AMS_ESEManagement Class

This class is the interface to access the executable software element information. Its operations allow:

- subscription to periodic state updates of some executable software elements,
- subscription to the change of state of some executable software elements, and
- access to other interfaces for a more specific view on elements.

Operations	*Parameters*	*Parameters type*
GetESE	return type	Collection<AMS_ExecutableSoftwareElement>

	filter	AMS_SWFilter
SubscribeESEStatusChange	return type	uint16
	filter	AMS_SWFilter
	out subscriptionID	uint32
SubscribeESEStatus	return type	uint16
	delay	uint16
	filter	AMS_SWFilter
	out subscriptionID	uint32
ShutdownESE	return type	uint16
	filter	AMS_SWFilter
GetESESpec	return type	Collection<AMS_ESESpec>
Unsubscribe	return type	uint16
	subscriptionID	uint32

The following sections review the operation in detail.

7.1.3.5.1 GetESE (filter: AMS_SWFilter)

This operation returns all executable software elements that match the specified filter.

7.1.3.5.2 GetESESpec ( )

This operation returns all executable software elements specifications.

7.1.3.5.3 ShutDownESE (filter: AMS_SWFilter)

This operation shuts down all executable software elements that match the specified filter.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

A full explanation of the cause of the error shall be logged.

7.1.3.5.4 SubscribeESEStatus (delay: uint16, filter: AMS_SWFilter, out subscriptionID : uint32)

This operation subscribes to receive periodically the status of the executable software elements matching with "filter."

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The effect of this operation is subscription to AMS_RTHWIndication class.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

7.1.3.5.5 SubscribeESEStatusChange (filter: AMS_SWFilter, out subscriptionID : uint32)

This operation subscribes to the modifications of the status of the executable software elements matching with "filter."

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The effect of this operation is subscription to AMS_RTHWIndication class.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

7.1.3.5.6 Unsubscribe (subscriptionID: uint32)

This operation deletes a previous subscription demand.

This operation shall return AMS_BADSUBSCRIPTIONID if the parameter is erroneous..

7.1.3.6 AMS_HWFilter Class

This class belongs to the "Hardware System Management" profile.

The AMS_HWFilter class models filters on hardware items.

The attributes are:

- A string containing a regular expression on the name of the hardware items (basic regular expression as described in [REGEXP]).
- Some hardware groups in which the hardware must be found.
- Some domains in which the hardware must be found.
- Some networks in which network links and network elements must be found.

Attributes	*Type*	
Filter	String	
Associations	*Multiplicity*	*Class*
AMS_GroupsInFilter	* .. *	AMS_HardwareGroup
AMS_DomainsInFilter	* .. *	AMS_Domain
AMS_NetworkInFilter	* .. *	CIM_Network

7.1.3.7 AMS_HWManagement Class

This class belongs to the "Hardware System Management" profile. This class is the interface to access the logical hardware information. Its operations allow:

- subscription to periodical state updates of some logical hardware,
- subscription to the change of state of some logical hardware, and
- access to other interfaces for a more specific view on elements.

Operations	*Parameters*	*Parameters type*
GetNetworkLinks	return type	Collection<CIM_ProtocolEndPoint>
	filter	AMS_HWFilter
SubscribeNetworkLoadChange	return type	uint16
	filter	AMS_HWFilter
	out subscriptionID	uint32
SubscribeNetworkLoad	return type	uint16
	delay	uint16
	filter	AMS_HWFilter
	out subscriptionID	uint32
CreateHardwareGroup	return type	uint16
	location	CIM_Location
	connectivity	String
	devices	String
	out group	AMS_HardwareGroup
GetAllLocations	return type	Collection<CIM_Location>
SubscribeHWStatusChange	return type	uint16
	filter	AMS_HWFilter
	out subscriptionID	uint32
SubscribeHWStatus	return type	uint16
	delay	uint16
	filter	AMS_HWFilter
	out subscriptionID	uint32
Unsubscribe	return type	uint16
	subscriptionID	uint32
GetComputerSystems	return type	Collection<AMS_ComputerSystem>
GetHardwareGroups	return type	Collection<AMS_HardwareGroup>
GetNetworks	return type	Collection<CIM_Network>
GetDomains	return type	Collection<AMS_Domain>

The following sections review the operation in detail.

7.1.3.7.1 CreateHardwareGroup (location: CIM_Location, connectivity: String, devices: String, out group: AMS_HardwareGroup)

This operation creates a hardware group from:

- a physical location
- subnets (parameter "connectivity")
- device types (parameter "devices")

It shall return:

- AMS_BADCONNECTIVITY if the 'connectivity' parameter is wrong.
- AMS_BADDEVICES if the 'devices' parameter is wrong,

7.1.3.7.2 GetAllLocations ( )

This operation returns all locations known by the AMSM service.

7.1.3.7.3 GetNetworkLinks (filter: AMS_HWFilter)

This operation returns all network links that match the specified filter.

7.1.3.7.4 SubscribeHWStatus (delay: uint16, filter: AMS_HWFilter, out subscriptionID : uint32)

This operation subscribes to receive periodically the status of the hardware items matching with "filter."

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The effect of this operation is subscription to AMS_RTHWIndication class.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

7.1.3.7.5 SubscribeHWStatusChange (filter: AMS_HWFilter, out subscriptionID : uint32)

This operation subscribes to the modifications of the status of the hardware items matching with "filter."

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The effect of this operation is subscription to AMS_RTHWIndication class.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

7.1.3.7.6 SubscribeNetworkLoad (delay: uint16, filter: AMS_HWFilter, out subscriptionID : uint32)

This operation subscribes to periodic updates of the network load elements matching with "filter."

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

This operations returns a collection of AMS_RTHWIndication.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

7.1.3.7.7 SubscribeNetworkLoadChange (filter: AMS_HWFilter, out subscriptionID : uint32)

This operation subscribes to the modifications of the load of the network elements matching with "filter."

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The effect of this operation is subscription to AMS_RTHWIndication class.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

7.1.3.7.8 Unsubscribe (subscriptionID: uint32)

This operation deletes a previous subscription demand.

This operation shall return AMS_BADSUBSCRIPTIONID if the parameter is erroneous.

7.1.3.7.9 GetComputerSystems(filter : AMS_HWFilter)

This operation returns all computers that match the specified filter.

7.1.3.7.10 GetHardwareGroups(filter : AMS_HWFilter)

This operation returns all hardware groups that match the specified filter.

7.1.3.7.11 GetNetworks(filter : AMS_HWFilter)

This operation returns all networks that match the specified filter.

7.1.3.7.12 GetDomains(filter : AMS_HWFilter)

This operation returns all domains that match the specified filter.

7.1.3.8 AMS_LoadBalancingManagement Class

This class belongs to the "Load Balancing Management" profile.

This class is the interface to access the load balancing group information. Its operations allow:

- subscription to periodic state updates of some load balancing groups,
- subscription to the change of state of some load balancing groups, and
- access to other interfaces for a more specific view on elements.

Operations	*Parameters*	*Parameters type*
GetLB	return type	Collection<AMS_LoadBalancingGroup>
	filter	AMS_SWFilter
SubscribeLBStatusChange	return type	uint16
	filter	AMS_SWFilter
	out subscriptionID	uint32

SubscribeLBStatus	return type	uint16
	delay	uint16
	filter	AMS_SWFilter
	out subscriptionID	uint32
Unsubscribe	return type	uint16
	subscriptionID	uint32

The following sections review the operation in detail.

7.1.3.8.1 GetLB (filter: AMS_SWFilter)

This operation returns all load balancing groups that match the specified filter.

7.1.3.8.2 SubscribeLBStatus (delay: uint16, filter: AMS_SWFilter, out subscriptionID : uint32)

This operation subscribes to receive periodically the status of the load balancing groups matching with "filter."

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The effect of this operation is subscription to AMS_RTHWIndication class.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

7.1.3.8.3 SubscribeLBStatusChange (filter: AMS_SWFilter, out subscriptionID : uint32)

This operation subscribes to the modifications of the status of the load balancing groups matching with "filter."

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

This operation returns a collection of AMS_RTHWIndication.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

7.1.3.8.4 Unsubscribe (subscriptionID: uint32)

This operation deletes a previous subscription demand.

This operation shall return AMS_BADSUBSCRIPTIONID if the parameter is erroneous.

7.1.3.9 AMS_Log Class

The class describes the log and its characteristics.

Since there is just one log per AMSM service, it is a singleton.

This class is an interface that permits to get the administrating and consuming interfaces of the normalized lightweight logging service.

Associations	*Multiplicity*	*Class*
CIM_LogManagesRecord {override}	* .. *	AMS_LogRecord
AMS_LogAdministrator	1 .. 1	LogAdministrator
AMS_LogConsumer	1 .. 1	LogConsumer

7.1.3.10 AMS_LogRecord Class

The AMS_LogRecord class is used to instantiate records to be aggregated to a Log. It is logically equivalent with a LogRecord of the Lightweight Logging Service.

Associations	*Multiplicity*	*Class*
CIM_LogicalIdentity	1 .. 1	LogRecord
CIM_LogManagesRecord {key} {override}	1 .. 1	AMS_Log

7.1.3.11 AMS_RTHWIndication Class

This class belongs to the "Hardware System Management" profile.

The AMS_RTHWIndication class gathers information on hardware status.

Issue 11404 - Non-coherent naming of some items in enumerates'

Its associations represent an AMS_ComputerSystem and an AMS_Property<AMS_StdHWUtilisation,HU_NONSTD>. The second is the status of the first.

Associations	*Multiplicity*	*Class*
AMS_NetworkElt	1 .. 1	AMS_ComputerSystem
AMS_RTHS	1 .. 1	AMS_Property<AMS_StdHWUtilisation,HU_NONSTD>

7.1.3.12 AMS_RTSWIndication Class

The AMS_RTSWIndication class gathers information on software status.

Its associations represent an AMS_ExecutableSoftwareElement or an AMS_Application or an AMS_SoftwareSystem or an AMS_LoadBalancingGroup or an AMS_RedundancyGroup, and an AMS_RTSoftwareStatus. The last is the status of one of the firsts.

Associations	*Multiplicity*	*Class*
AMS_ESE	0 .. 1	AMS_ExecutableSoftwareElement
AMS_AppIndication	0 .. 1	AMS_Application

AMS_System	0 .. 1	AMS_SoftwareSystem
AMS_LB	0 .. 1	AMS_LoadBalancingGroup
AMS_RG	0 .. 1	AMS_RedundancyGroup
AMS_RTSW	1 .. 1	AMS_RTSoftwareStatus

7.1.3.13 AMS_RedundancyGroupManagement Class

This class belongs to the "Fault Tolerance Management" profile. This class is the interface to access the redundancy group information. Its operations allow:

- subscription to periodical state updates of some redundancy groups,
- subscription to the change of state of some redundancy groups, and
- access to other interfaces for a more specific view on elements.

Operations	*Parameters*	*Parameters type*
GetRG	return type	Collection<AMS_RedundancyGroup>
	filter	AMS_SWFilter
SubscribeRGStatusChange	return type	uint16
	filter	AMS_SWFilter
	out subscriptionID	uint32
SubscribeRGStatus	return type	uint16
	delay	uint16
	filter	AMS_SWFilter
	out subscriptionID	uint32
Unsubscribe	return type	uint16
	subscriptionID	uint32

The following sections review the operation in detail.

7.1.3.13.1 GetRG (filter: AMS_SWFilter)

This operation returns all redundancy groups that match the specified filter.

7.1.3.13.2 SubscribeRGStatus (delay: uint16, filter: AMS_SWFilter, out subscriptionID : uint32)

This operation subscribes to receive periodically the status of the redundancy groups matching with "filter."

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

This operation returns a collection of AMS_RTHWIndication.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

7.1.3.13.3 SubscribeRGStatusChange (filter: AMS_SWFilter, out subscriptionID : uint32)

This operation subscribes to the modifications of the status of the redundancy groups matching with "filter."

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The effect of this operation is subscription to AMS_RTHWIndication class.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

7.1.3.13.4 Unsubscribe (subscriptionID: uint32)

This operation deletes a previous subscription demand.

This operation shall return AMS_BADSUBSCRIPTIONID if the parameter is erroneous.

7.1.3.14 AMS_SAMManagement Class

This class is the interface to access the Supported Applications Models. Its operations allow:

- an access to other interfaces for a more specific view on elements.

Operations	*Parameters*	*Parameters type*
GetAllSAM	return type	Collection<AMS_SupportedApplicationManagement>

The following sections review the operation in detail.

7.1.3.14.1 GetAllSAM ( )

This operation returns all application models known by the AMSM service.

7.1.3.15 AMS_SWFilter Class

The AMS_SWFilter class models filters on software items.

The attributes or associations are:

- A string containing a regular expression on the name of the software items and/or a logical expression on attributes of ESE, applications or system (basic regular expression as described in [REGEXP]).
- A list of required states.

Attributes	*Type*	
Filter	String	
Associations	*Multiplicity*	*Class*
StateFilters	0 .. *	AMS_State

7.1.3.16 AMS_SystemManagement Class

This class is the interface to access the system information. Its operations allow:

- subscription to periodic state updates of some systems,
- subscription to the change of state of some systems, and
- access to other interfaces for a more specific view on elements.

Operations	*Parameters*	*Parameters type*
GetSystem	return type	Collection<AMS_SoftwareSystem>
	filter	AMS_SWFilter
SubscribeSystemStatusChange	return type	uint16
	filter	AMS_SWFilter
	out subscriptionID	uint32
SubscribeSystemStatus	return type	uint16
	delay	uint16
	filter	AMS_SWFilter
	out subscriptionID	uint32
Unsubscribe	return type	uint16
	subscriptionID	uint32

The following sections review the operation in detail.

7.1.3.16.1 GetSystem (filter: AMS_SWFilter)

This operation returns all software systems that match the specified filter.

7.1.3.16.2 SubscribeSystemStatus (delay: uint16, filter: AMS_SWFilter, out subscriptionID : uint32)

This operation subscribes to receive periodically the status of the software systems matching with "filter."

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The effect of this operation is subscription to AMS_RTHWIndication class.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

7.1.3.16.3 SubscribeSystemStatusChange (filter: AMS_SWFilter, out subscriptionID : uint32)

This operation subscribes to the modifications of the status of the software systems matching with "filter".

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The data returned are a collection of AMS_RTHWIndication.

This operation shall return AMS_BADFILTER if the filter parameter is wrong.

7.1.3.16.4 Unsubscribe (subscriptionID: uint32)

This operation deletes a previous subscription demand.

This operation shall return AMS_BADSUBSCRIPTIONID if the parameter is erroneous.

7.1.4 Application Package

The "Application" package groups the classes needed to manage and monitor applications while they are running. The information needed to define (i.e., before runtime) applications has been put in the "Application Specification" package. Since some items needed to manage applications are defined beforehand, there are links from this package to the "Application Specification" package.

This package does not introduce the deployment aspects, which are included in the "Application Deployment" package.

Roughly, an application (AMS_Application) is designed as a set of executable software elements and/or redundancy groups and/or load balanced groups. A redundancy group (AMS_RedundancyGroup) gathers executable software elements which are executed in a redundant way. A load balancing group (AMS_LoadBalancingGroup) gathers executable software elements, which are executed in a load, balanced way.

Each of these elements:

- Is linked to a specification (AMS_ESESpec or AMS_SoftwareFeatureSpec), which stores:
  - the information which were used for the creation of the element - when deploying the software element specification, and
  - the information which will be needed later on for management (start, stop, etc.).
- Recognizes some management and monitoring interfaces.

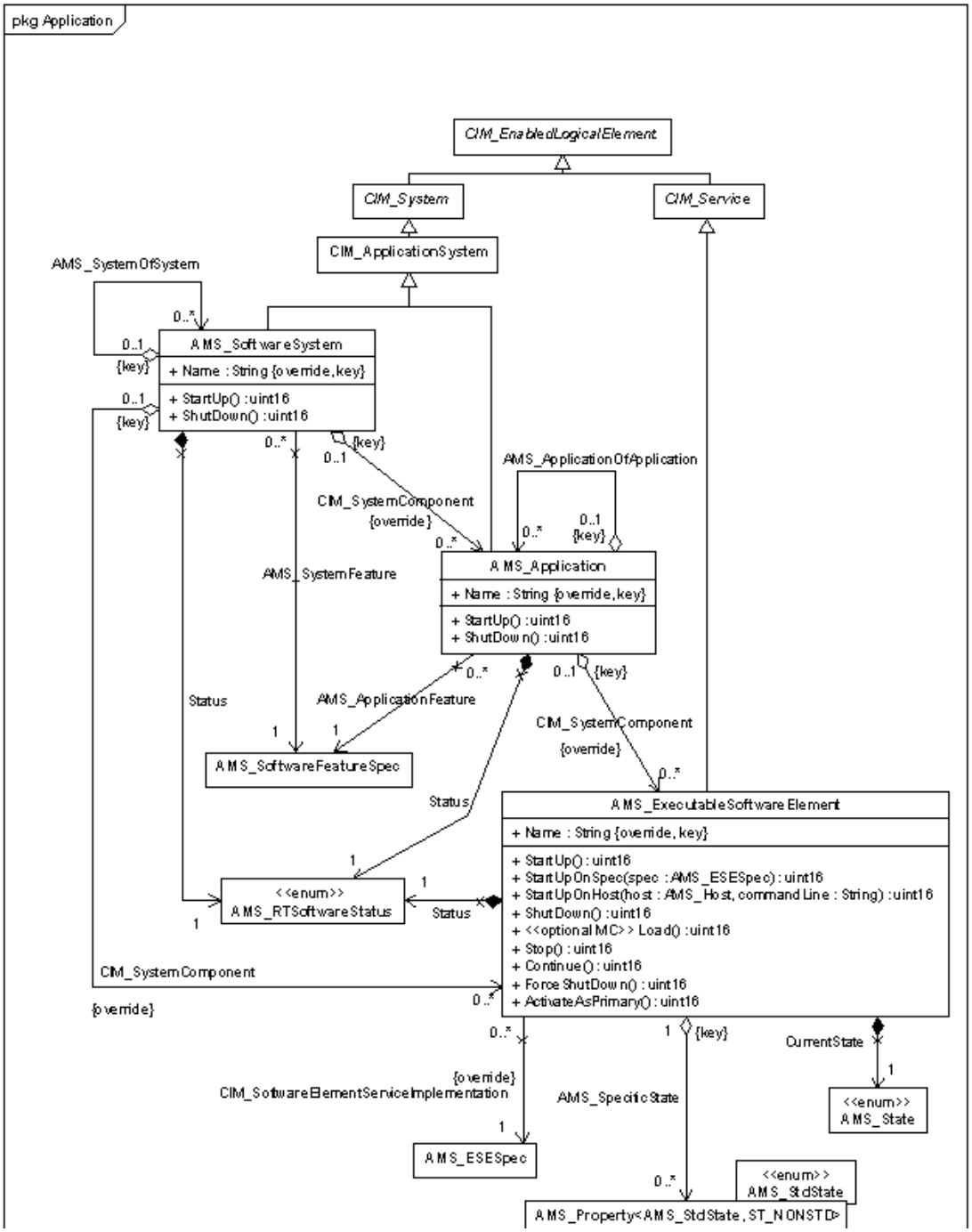


Figure 7.7 - Application class diagram (1/3)

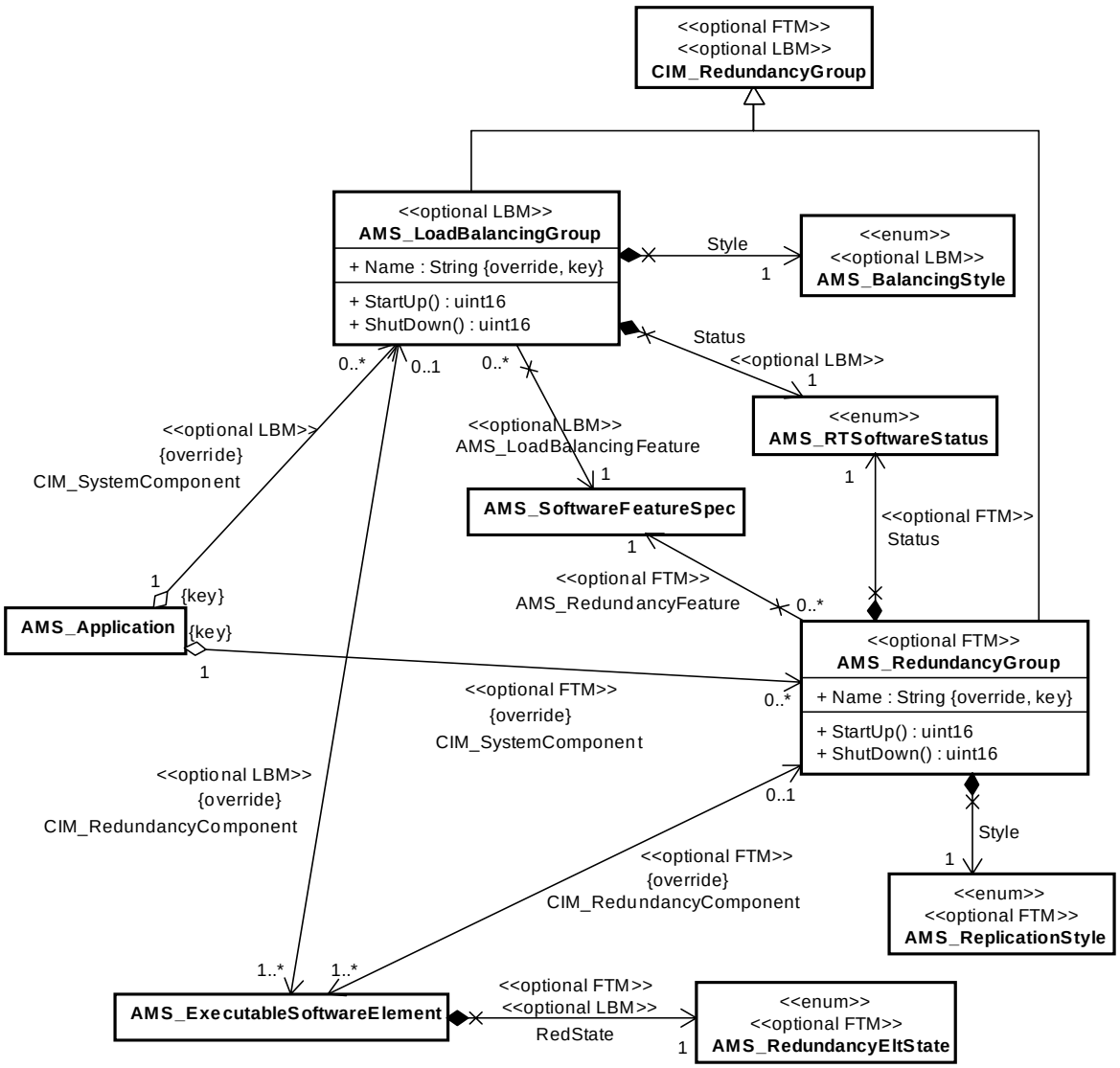


Figure 7.8 - Application Class diagram (2/3)

Moreover process and thread are designed through the CIM_Process and CIM_Thread classes. When a Unix operating system is involved, the CIM_UnixProcess and CIM_UnixThread classes are to be used.

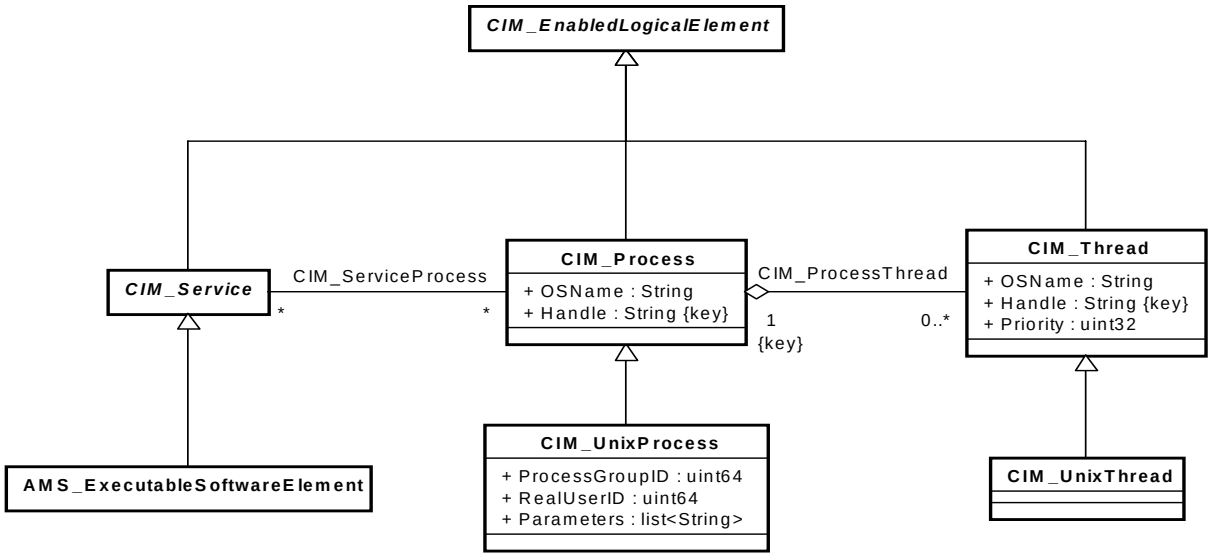


Figure 7.9 - Application class diagram (3/3)

The following instantiation diagram shows an example of such elements.

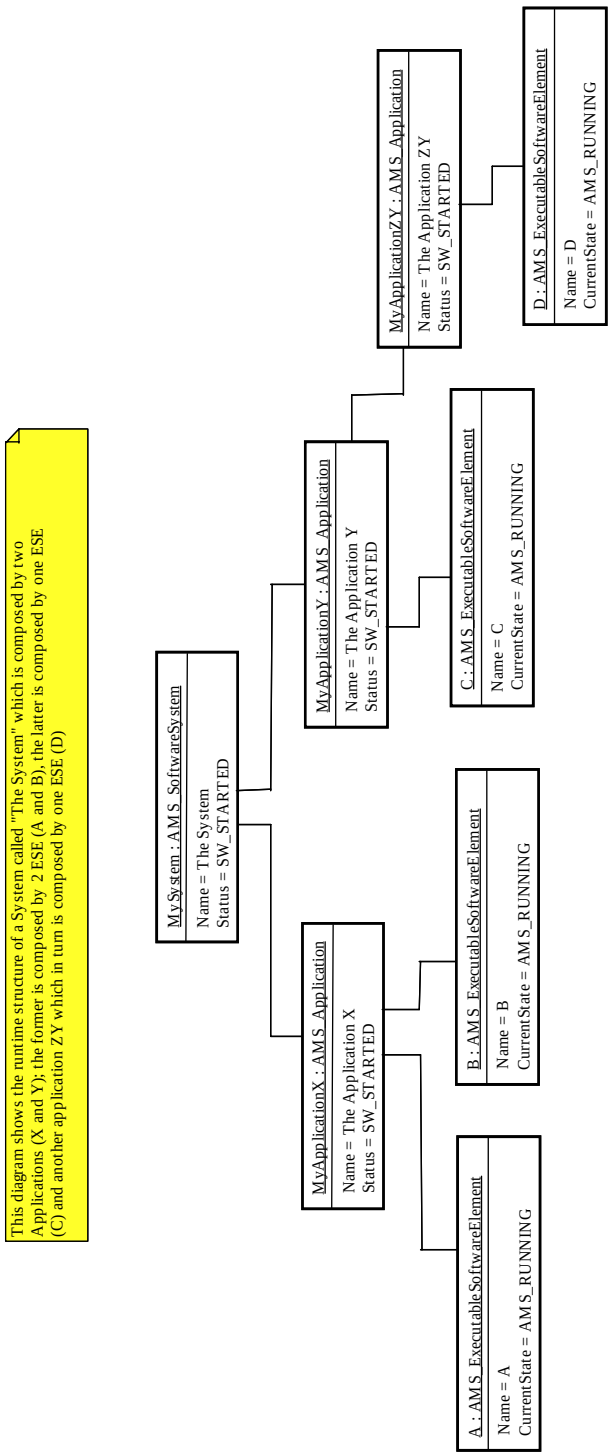


Figure 7.10 - Application instantiation

And the following sequence diagram shows an example of use of the API.

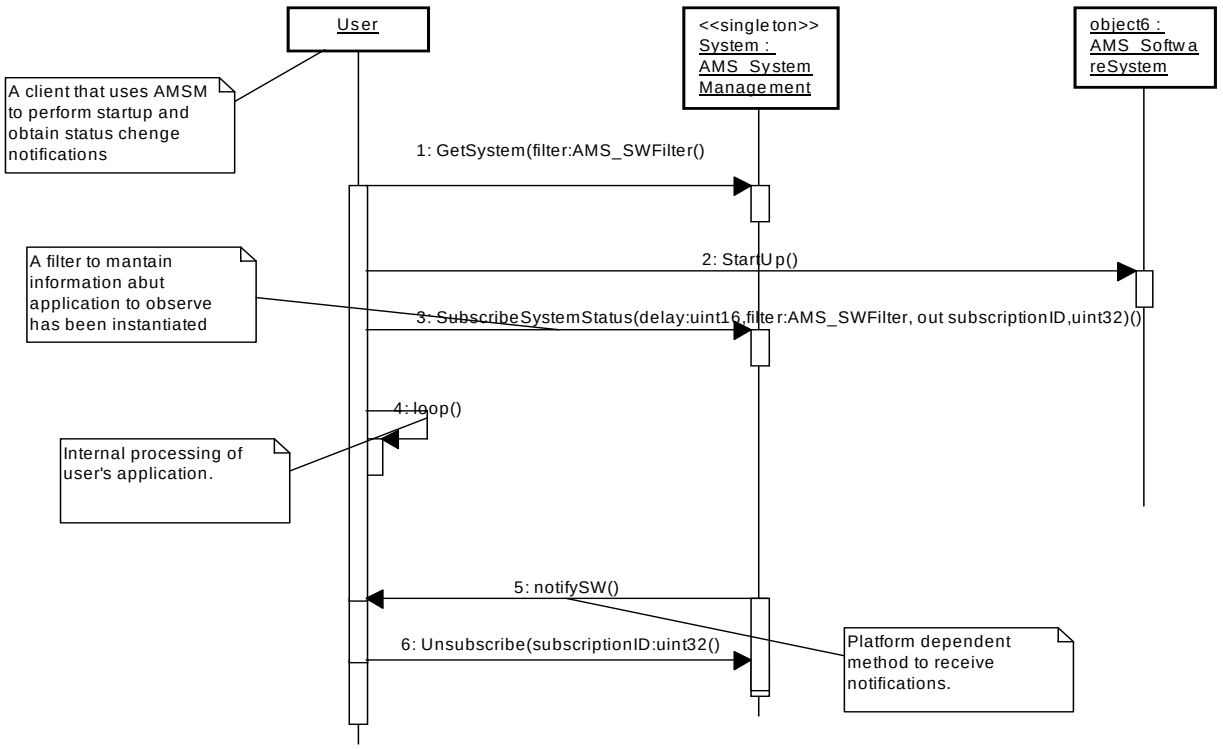


Figure 7.11 - Exemple of application start sequence

7.1.4.1 AMS_Application Class

The AMS_Application class models the applications.

It is a part of an AMS_SoftwareSystem, may gather AMS_ExecutableSoftwareElements, AMS_LoadBalancingGroups, and AMS_RedundancyGroups and must be associated to the AMS_SoftwareFeatureSpec that has been used to create it and which refers to creation information.

It may also gather other AMS_Applications with the association "AMS_ApplicationOfApplication."

Its attributes are:

- A name which, with its CIM_SoftwareSystem's global name, takes part of the make-up of its global name.
- A status (started, stopped, or failed).

An AMS_Application is an interface that allows to start, stop, and get the status of the application itself.

Attributes	*Type*	
Name {override, key}	String	
Associations	*Multiplicity*	*Class*
CIM_SystemComponent {override}	0 .. *	AMS_ExecutableSoftwareElement

CIM_SystemComponent {key} {override}	0 .. 1	AMS_SoftwareSystem
AMS_ApplicationFeature	1 .. 1	AMS_SoftwareFeatureSpec
CIM_SystemComponent {override} <<optional FTM>>	0 .. *	AMS_RedundancyGroup
AMS_ApplicationOfApplication {key}	0 .. 1	AMS_Application
AMS_ApplicationOfApplication	0 .. *	AMS_Application
CIM_SystemComponent {override} <<optional LBM>>	0 .. *	AMS_LoadBalancingGroup
Status	1 .. 1	AMS_RTSoftwareStatus
Operations	*Parameters*	*Parameters type*
StartUp	return type	uint16
ShutDown	return type	uint16

The following sections review the operation in detail.

7.1.4.1.1 ShutDown ( )

This operation shuts down the application. It shuts down all the executable software elements which belong to the application.

It shall return AMS_SHUTDOWNFAILED if at least one of the executable software elements could not shutdown. The state of the other executable software elements is then undefined.

A full explanation of the cause of the error shall be logged.

After an error, the executable software elements are in an unknown state.

7.1.4.1.2 StartUp ( )

This operation starts up the application. It starts up all the executable software elements which belong to the application.

It shall return AMS_STARTFAILED if at least one of the executable software element could not start. The state of the other executable software elements is the undefined.

A full explanation of the cause of the error shall be logged.

After an error, the executable software elements are in an unknown state.

Issue 11613 - Paragraph 7.1.5 should be renumbered 7.1.4.2

7.1.4.2 AMS_BalancingStyle Class

This class belongs to the "Load Balancing Management" profile.

The AMS_BalancingStyle class enumerates possible style for balancing in a load balancing group.

Possible values are:

- “Round Robin:” If a request from a client is balanced with a Round-Robin strategy upon a group of members, the exact order in which requests are handed over to group members may be implementation-dependent; however, the service must guarantee that, for a group with n members, if a request is forwarded to a particular group member, the next n-1 requests, from the same client, are not forwarded to that member.
- “Random:” Requests are handed over to group members in a randomly way.
- “Implementation Defined:” The exact order in which requests are handed over to group members is defined by the implementation.

Enumeration
LB_ROUND_ROBIN
LB_RANDOM
LB_IMPL_DEFINED

7.1.4.3 AMS_ExecutableSoftwareElement Class

The AMS_ExecutableSoftwareElement class models a software element that is an executable program or equivalent (script, .jar file, single executable CORBA component, etc.) that will be directly managed (i.e., started, stopped, configured, and monitored) by the application management service. This does not include elements such as libraries, which are not executed by themselves. It is part of an AMS_Application, may be associated with an AMS_RedundancyGroup, and must be associated to the AMS_ESESpec that has been used to create it and retains most of the information on the element.

Its attributes and associations are:

- A name that, with its CIM_Application’s or AMS_Application’s global name, takes part of the make-up of its global name.
- A current state that is taken in the states allowed by its application model.
- A status (started, stopped, or failed).
- A state regarding the redundancy (no redundancy, primary or redundant copy).

Issue 11405 - Typo 1

- A platform-specific states designed by with the AMS_Property<AMS_StdState, ST_NONSTD> instanciation of the AMS_Property template. The AMS_StdState enumeration gives some normalized types of state.

Note – Other well-known platform-specific states such as PID, GID, and so forth are designed in the CIM_Process and CIM_Thread classes.

An AMS_Application is an interface that allows starting, stopping, loading, continuing, and getting the status of the software element itself.

Attributes	*Type*	
Name {override, key}	String	
Associations	*Multiplicity*	*Class*
CIM_SystemComponent {key} {override}	0 .. 1	AMS_Application
CIM_SoftwareElementServiceImplementation {override}	1 .. 1	AMS_ESESpec
CurrentState	1 .. 1	AMS_State
CIM_SystemComponent {key} {override}	0 .. 1	AMS_SoftwareSystem
CIM_RedundancyComponent {override} <<optional FTM>>	0 .. 1	AMS_RedundancyGroup
CIM_RedundancyComponent {override} <<optional LBM>>	0 .. 1	AMS_LoadBalancingGroup
Status	1 .. 1	AMS_RTSoftwareStatus
RedState <<optional FTM>> <<optional LBM>>	1 .. 1	AMS_RedundancyEltState
AMS_ESEDeployed	1 .. 1	AMS_DeploymentLink
AMS_SpecificState	0 .. *	AMS_Property<AMS_StdState,ST_NONSTD>
Operations	*Parameters*	*Parameters type*
StartUp	return type	uint16
StartUpOnSpec	return type	uint16
	spec	AMS_ESESpec
StartUpOnHost	return type	uint16
	host	AMS_Host
	commandLine	String
ShutDown	return type	uint16
Load	return type	uint16
Stop	return type	uint16

Continue	return type	uint16
ForceShutDown	return type	uint16
ActivateAsPrimary	return type	uint16

The following sections review the operation in detail.

7.1.4.3.1 ActivateAsPrimary ( )

This operation activates the executable software element such as it becomes the primary element in its redundancy group. It shall return:

- AMS_ALREADYPRIMARY if the element was already a primary element.
- AMS_NOTFT if the element is not in a redundancy group.
- AMS_PRIMARYFAILED if the element could not be activated for any other reason.

In all cases of error, a full explanation of the cause of the error shall be logged.

This method does not affect the state of the executable software element.

7.1.4.3.2 Continue ( )

This operation continues the executable software element. It shall return:

- AMS_RESOURCEERROR if the element could not be continued on account of an OS resource starvation error (lack of memory, disk) when performing one of the actions of the associated specification.
- AMS_RIGHTERROR if the element could not be continued on account of an OS right error when performing one of the actions of the associated specification.
- AMS_BADSTATE if the element is not in a state in which it could be continued.
- AMS_CONTFAILED if the element could not be continued for any other reason.

In all case of error, a full explanation of the cause of the error shall be logged.

After an error (but AMS_BADSTATE), the executable software element is in the ERROR state.

7.1.4.3.3 ForceShutDown ( )

This operation forces the shutdown of the executable software element. It shall return:

- AMS_RESOURCEERROR if the element could not be shut down on account of an OS resource starvation error (lack of memory, disk) when performing one of the actions of the associated specification.
- AMS_RIGHTERROR if the element could not be shut down on account of an OS right error when performing one of the actions of the associated specification.
- AMS_BADSTATE if the element is not in a state in which it could be shut down.
- AMS_SHUTDOWNFAILED if the element could not be shut down for any other reason.

In all case of error, a full explanation of the cause of the error shall be logged.

After an error (but AMS_BADSTATE), the executable software element is in the ERROR state.

7.1.4.3.4 Load ( )

This operation loads the executable software element into memory. This operation is optional (“Maximum Control” profile). It shall return:

- AMS_RESOURCEERROR if the element could not start on account of an OS resource starvation error (lack of memory, disk) when performing one of the actions of the associated specification.
- AMS_RIGHTERROR if the element could not start on account of an OS right error when performing one of the actions of the associated specification.
- AMS_BADMODELTYPE if the targeted host does not support the associated specification model type.
- AMS_BADSTATE if the element is not in a state in which it could be loaded.
- AMS_LOADFAILED if the element could not start for any other reason.

In all case of error, a full explanation of the cause of the error shall be logged.

After an error (but AMS_BADSTATE), the executable software element is in the ERROR state.

7.1.4.3.5 ShutDown ( )

This operation shuts down the executable software element. It shall return:

- AMS_RESOURCEERROR if the element could not be shut down on account of an OS resource starvation error (lack of memory, disk) when performing one of the actions of the associated specification.
- AMS_RIGHTERROR if the element could not be shut down on account of an OS right error when performing one of the actions of the associated specification.
- AMS_BADACTION if the element could not be shut down on account of a badly formed action in the specification definition.
- AMS_BADCHECK if the element could not be shut down on account of a badly formed check in the specification definition.
- AMS_NOTCHECKED if at least one of the checks of the associated specification was not true.
- AMS_BADSTATE if the element is not in a state in which it could be shut down.
- AMS_SHUTDOWNFAILED if the element could not be shut down for any other reason.

In all case of error, a full explanation of the cause of the error shall be logged.

After an error (but AMS_BADSTATE), the executable software element is in the ERROR state.

7.1.4.3.6 StartUpOnHost (host: AMS_Host, commandLine: String)

This operation starts up the executable software element on the host and with the command line indicated by the parameters. It shall return:

- AMS_RESOURCEERROR if the element could not start on account of an OS resource starvation error (lack of memory, disk) when performing one of the actions of the associated specification.

- AMS_RIGHTERROR if the element could not start on account of an OS right error when performing one of the actions of the associated specification.
- AMS_BADACTION if the element could not start on account of a badly formed action in the specification definition in parameter.
- AMS_BADCHECK if the element could not start on account of a badly formed check in the specification definition in parameter.
- AMS_NOTCHECKED if at last one of the checks of the specification in parameter was not true.
- AMS_BADMODELTYPE if the targeted host does not support the associated specification model type.
- AMS_BADCOMMANDLINE if the command line in parameter is wrong.
- AMS_BADSTATE if the element is not in a state in which it could be started.
- AMS_STARTFAILED if the element could not start for any other reason.

In all case of error, a full explanation of the cause of the error shall be logged.

After an error (but AMS_BADSTATE), the executable software element is in the ERROR state.

7.1.4.3.7 StartUpOnSpec (spec: AMS_ESESpec)

This operation replaces the AMS_ESESpec associated by CIM_SoftwareElementServiceImplementation, and next starts up the executable software element.

It allows to change at run time the way an ESE is started, shutdowned, etc. by changing its actions and checks set. It shall return:

- AMS_RESOURCEERROR if the element could not start on account of an OS resource starvation error (lack of memory, disk) when performing one of the action of the associated specification.
- AMS_RIGHTERROR if the element could not start on account of an OS right error when performing one of the action of the associated specification.
- AMS_BADACTION if the element could not start on account of a badly formed action in the specification definition in parameter.
- AMS_BADCHECK if the element could not start on account of a badly formed check in the specification definition in parameter.
- AMS_NOTCHECKED if at last one of the check of the specification in parameter was not true.
- AMS_BADMODELTYPE if the targeted host does not support the associated specification model type.
- AMS_BADSTATE if the element is not in a state in which it could be started.
- AMS_STARTFAILED if the element could not start for any other reason.

In all case of error, a full explanation of the cause of the error shall be logged.

After an error (but AMS_BADSTATE), the executable software element is in the ERROR state.

7.1.4.3.8 StartUp ( )

This operation starts up the executable software element with the specification actually associated by CIM_SoftwareElementServiceImplementation. It shall return:

- AMS_RESOURCEERROR if the element could not start on account of an OS resource starvation error (lack of memory, disk) when performing one of the actions of the associated specification.
- AMS_RIGHTERROR if the element could not start on account of an OS right error when performing one of the actions of the associated specification.
- AMS_BADACTION if the element could not start on account of a badly formed action in the specification definition.
- AMS_BADCHECK if the element could not start on account of a badly formed check in the specification definition.
- AMS_NOTCHECKED if at last one of the check of the associated specification was not true.
- AMS_BADMODELTYPE if the targeted host does not support the associated specification model type.
- AMS_BADSTATE if the element is not in a state in which it could be started.
- AMS_STARTFAILED if the element could not start for any other reason.

In all case of error, a full explanation of the cause of the error shall be logged.

After an error (but AMS_BADSTATE), the executable software element is in the ERROR state.

7.1.4.3.9 Stop ( )

This operation stops the executable software element. It shall return:

- AMS_RESOURCEERROR if the element could not be stopped on account of an OS resource starvation error (lack of memory, disk) when performing one of the actions of the associated specification.
- AMS_RIGHTERROR if the element could not be stopped on account of an OS right error when performing one of the actions of the associated specification.
- AMS_BADSTATE if the element is not in a state in which it could be stopped.
- AMS_STOPFAILED if the element could not be stopped for any other reason.

In all case of error, a full explanation of the cause of the error shall be logged.

After an error (but AMS_BADSTATE), the executable software element is in the ERROR state.

7.1.4.4 AMS_LoadBalancingGroup Class

This class belongs to the "Load Balancing Management" profile.

The AMS_LoadBalancingGroup class models the groups of elements that are load balanced.

Load Balancing service provides the ability to optimize the distribution of load among the available servers of the system. In this context the concept of "Strategy" means the rule used by each application for choosing the server to execute the request within the available replicas.

It is a part of an AMS_Application, must be associated to the AMS_ExecutableElements that belong to it, and must be associated to the AMS_SoftwareFeatureSpec that has been used to create it and which stores creation information.

Its attributes and associations are:

- A name which, with its CIM_Application's global name, takes part of the make-up of its global name.
- A style which is the strategy to be used for that group.
- A status (started, stopped, or failed).

An AMS_LoadBalancingGroup is an interface that allows to start, stop and retrieve the status of the group.

Attributes	*Type*	
Name {override, key}	String	
Associations	*Multiplicity*	*Class*
CIM_SystemComponent {key} {override}	1 .. 1	AMS_Application
AMS_LoadBalancingFeature	1 .. 1	AMS_SoftwareFeatureSpec
CIM_RedundancyComponent {override}	1 .. *	AMS_ExecutableSoftwareElement
Style	1 .. 1	AMS_BalancingStyle
Status	1 .. 1	AMS_RTSoftwareStatus
Operations	*Parameters*	*Parameters type*
StartUp	return type	uint16
ShutDown	return type	uint16

The following sections review the operation in detail.

7.1.4.4.1 ShutDown ( )

This operation shuts down the load balancing group. It must stop down all the executable software elements which belong to the group.

It shall return AMS_SHUTDOWNFAILED if at least one of the executable software element could not shutdown. The state of the other executable software elements is the undefined.

A full explanation of the cause of the error shall be logged.

7.1.4.4.2 StartUp ( )

This operation starts up the load balancing group. It starts up all the executable software elements which belong to the group.

It shall return AMS_STARTFAILED if at least one of the executable software elements could not start. The state of the other executable software elements is the undefined.

A full explanation of the cause of the error shall be logged.

7.1.4.5 AMS_RTSoftwareStatus Class

The AMS_RTSoftwareStatus class enumerates run-time status of software items.

Enumeration
SW_STARTED
SW_STOPPED
SW_FAILED

7.1.4.6 AMS_RedundancyGroup Class

This class belongs to the "Fault Tolerance Management" profile.

The AMS_RedundancyGroup class models the groups of elements that are working in a redundant way.

It is a part of an AMS_Application, must be associated to the AMS_ExecutableElement that belongs to it, and must be associated to the AMS_SoftwareFeatureSpec that has been used to create it and which stores creation information.

Its attributes and associations are:

- A name which, with its CIM_Application's global name, takes part of the make-up of its global name.
- The style of replication (AMS_ReplicationStyle).
- A status (started, stopped, or failed).

An AMS_RedundancyGroup is an interface that allows to start, stop, and retrieve the status of the group.

Attributes	*Type*	
Name {override, key}	String	
Associations	*Multiplicity*	*Class*
CIM_RedundancyComponent {override}	1 .. *	AMS_ExecutableSoftwareElement
CIM_SystemComponent {key} {override}	1 .. 1	AMS_Application
AMS_RedundancyFeature	1 .. 1	AMS_SoftwareFeatureSpec
Style	1 .. 1	AMS_ReplicationStyle
Status	1 .. 1	AMS_RTSoftwareStatus
Operations	*Parameters*	*Parameters type*

StartUp	return type	uint16
ShutDown	return type	uint16

The following sections review the operation in detail.

7.1.4.6.1 ShutDown ( )

This operation shuts down the redundancy group. It must stops down all the executable software elements which belong to the group.

It shall return AMS_SHUTDOWNFAILED if at least one of the executable software elements could not shutdown. The state of the other executable software elements is the undefined.

A full explanation of the cause of the error shall be logged.

7.1.4.6.2 StartUp ( )

This operation starts up the redundancy group. It must start up all the executable software elements which belong to the group.

It shall return AMS_STARTFAILED if at least one of the executable software elements could not start. The state of the other executable software elements is the undefined.

A full explanation of the cause of the error shall be logged.

7.1.4.7 AMS_ReplicationStyle Class

This class belongs to the "Fault Tolerance Management" profile.

The AMS_ReplicationStyle class enumerates possible style for replication of a redundancy group.

This enumeration comes from Fault Tolerant CORBA ([CORBA]).

- RG_STATELESS: the object contains read only data, so there is no need for recording or transferring object's state.
- RG_COLD_PASSIVE: replicas are not loaded into memory and they only come into existence when the primary replica fails. Since there is only one primary replica at any one time, the primary replica's state must be captured in case it fails. If the primary replica fails, one of the cold backup replicas is loaded into memory, and assumes the role of the new primary replica. For the new primary replica to take over from the old primary replica, the new replica's state must be identical to the state of the old primary replica. Before the new primary can fully assume the role of the primary replica, its state is initialized using the last checkpoint recorded previously by the logging-recovery mechanisms.
- RG_WARM_PASSIVE: this replication style differs from the Cold Passive replication in that the state of the primary member object of the object group gets recorded and transferred to other member objects of the object group (i.e., backup replicas). This type of recovery provides faster recovery from faults than Cold Passive.
- RG_ACTIVE and RG_ACTIVE_WITH_VOTING: with this replication style all members of the object group execute the invoked methods simultaneously and expected to provide rapid recovery from faults.
- RG_IMPL_DEFINED: specific implementation defined style of replication.

Enumeration

RG_COLD_PASSIVE
RG_WARM_PASSIVE
RG_ACTIVE
RG_ACTIVE_WITH_VOTING
RG_STATELESS
RG_IMPL_DEFINED

7.1.4.8 AMS_SoftwareSystem Class

The AMS_SoftwareSystem class models the software systems.

It may be a part of another AMS_SoftwareSystem, gathers AMS_ExecutableSoftwareElements and AMS_Applications and must be associated to the AMS_SoftwareFeatureSpec that has been used to create it and which store creation information.

Its attributes and associations are:

- A name which, with its CIM_SoftwareSystem's global name, takes part of the make-up of its global name.
- A status (started, stopped, or failed).

An AMS_Application is an interface that allows to start, stop, and retrieve the status of the software system.

Attributes	*Type*	
Name {override, key}	String	
Associations	*Multiplicity*	*Class*
CIM_SystemComponent {override}	0 .. *	AMS_Application
AMS_SystemOfSystem	0 .. *	AMS_SoftwareSystem
AMS_SystemOfSystem {key}	0 .. 1	AMS_SoftwareSystem
AMS_SystemFeature	1 .. 1	AMS_SoftwareFeatureSpec
CIM_SystemComponent {override}	0 .. *	AMS_ExecutableSoftwareElement
Status	1 .. 1	AMS_RTSoftwareStatus
Operations	*Parameters*	*Parameters type*
StartUp	return type	uint16
ShutDown	return type	uint16

The following sections review the operation in detail.

7.1.4.8.1 ShutDown ( )

This operation shuts down the software system. It must stop down all the executable software elements which belong to the system.

It shall return AMS_SHUTDOWNFAILED if at least one of the executable software elements could not shutdown. The state of the other executable software elements is the undefined.

A full explanation of the cause of the error shall be logged.

7.1.4.8.2 StartUp ( )

This operation starts up the software system. It must start up all the executable software elements which belong to the system.

It shall return AMS_STARTFAILED if at least one of the executable software element could not start. The state of the other executable software elements is the undefined.

A full explanation of the cause of the error shall be logged.

7.1.5 Application Deployment Package

The "Application Deployment" package groups the classes needed to describe a deployment configuration while applications are running.

The information needed to define (before runtime) deployment has been put in the "Application Deployment Specification" package.

Since some items needed to manage deployments are defined beforehand, there are links from this package to the "Application Deployment Specification" package so as to store this information.

This package does not introduce either application or hardware aspects, which are elaborated in the "Application" and "Hardware" packages.

Generally, a deployment (AMS_DeploymentConfiguration) contains a set of deployment links (AMS_DeploymentLink) that defines the hosts that an application, a software system, or a software element are deployed.

The AMS_DeploymentConfiguration class:

- Is linked to a specification (AMS_DeploymentConfigurationSpec), which stores:
  - the information which were used for the creation of the element when deploying the deployment specification, and
  - the information which will be needed later on for management (start, shutdown).
- Recognizes some management and monitoring interfaces.

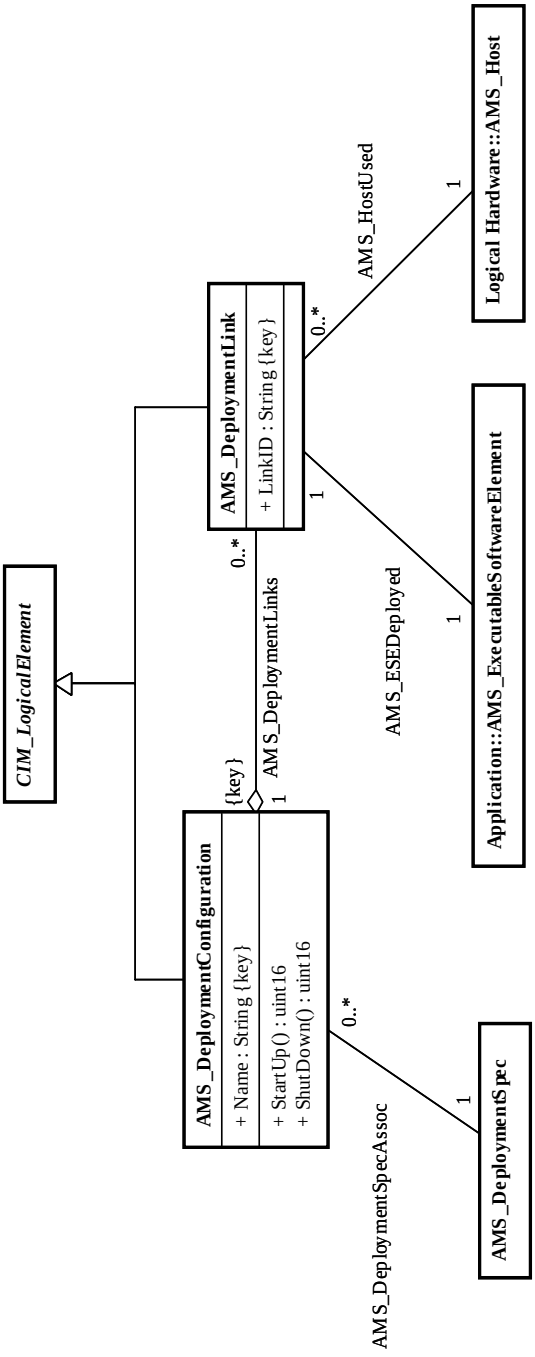


Figure 7.12 - Application Deployment class diagram

7.1.5.1 AMS_DeploymentConfiguration Class

The AMS_DeploymentConfiguration class models the deployment configurations (run time concept).

It gathers AMS_DeploymentLinks and must be associated to the AMS_DeploymentConfigurationSpec that has been used to create it and which store creation information.

Its attribute is the name of the deployment configuration.

An AMS_DeploymentConfiguration is an interface that allows to start or stop a deployment configuration.

Attributes	*Type*	
Name {key}	String	
Associations	*Multiplicity*	*Class*
AMS_DeploymentLinks	0 .. *	AMS_DeploymentLink
AMS_DeploymentSpecAssoc	1 .. 1	AMS_DeploymentSpec
Operations	*Parameters*	*Parameters type*
StartUp	return type	uint16
ShutDown	return type	uint16

The following sections review the operation in detail.

7.1.5.1.1 ShutDown ( )

This operation shuts down the deployment configuration: it must shut down all the executable software elements which belong to the deployment configuration.

It shall return AMS_SHUTDOWNFAILED if at least one of the executable software element could not shutdown. The state of the other executable software elements is then undefined.

A full explanation of the cause of the error shall be logged.

7.1.5.1.2 StartUp ( )

This operation starts up the deployment configuration: it must start up all the executable software elements which belong to the deployment configuration.

It shall return AMS_STARTFAILED if at least one of the executable software elements could not start. The state of the other executable software elements is the undefined.

A full explanation of the cause of the error shall be logged.

7.1.5.2 AMS_DeploymentLink Class

The AMS_DeploymentLink class models the fact that software (application, software system, or software element) is running on some hosts.

It belongs to an AMS_DeploymentConfiguration and is linked to some AMS_Hosts and an AMS_ExecutableSoftwareElement.

If the association with an AMS_ExecutableSoftwareElement is used, just one host is allowed.

Its attribute is a "LinkID" which, with its AMS_DeploymentConfiguration's global name, takes part of the make-up of its global name.

Attributes	*Type*	
LinkID {key}	String	
Associations	*Multiplicity*	*Class*
AMS_HostUsed	1 .. 1	AMS_Host
AMS_ESEDeployed	1 .. 1	AMS_ExecutableSoftwareElement
AMS_DeploymentLinks {key}	1 .. 1	AMS_DeploymentConfiguration

7.1.6 Application Deployment Specification Package

The "Application Deployment Specification" package groups the classes needed to model a deployment configuration so that they can be deployed subsequently. This package is a configuration view of deployment configurations.

The information needed to manage and monitor applications have been put in the "Application Deployment" package. The aim of the classes of this package is to describe the connections amongst hardware elements and software elements intended for future deployment. So, an AMS_DeploymentSpec is a set of links (AMS_DeploymentLinkSpec) that describes these connections.

These links allow the definition of the deployments:

- of:
  - either an executable software element specified by an AMS_ESESpec, or
  - another link form another deployment spec,
- on an actual host (AMS_Host).

In the case of the deployment of an executable software element, actions to be taken while starting, shutting down can be added to those defined on the executable software element specification itself.

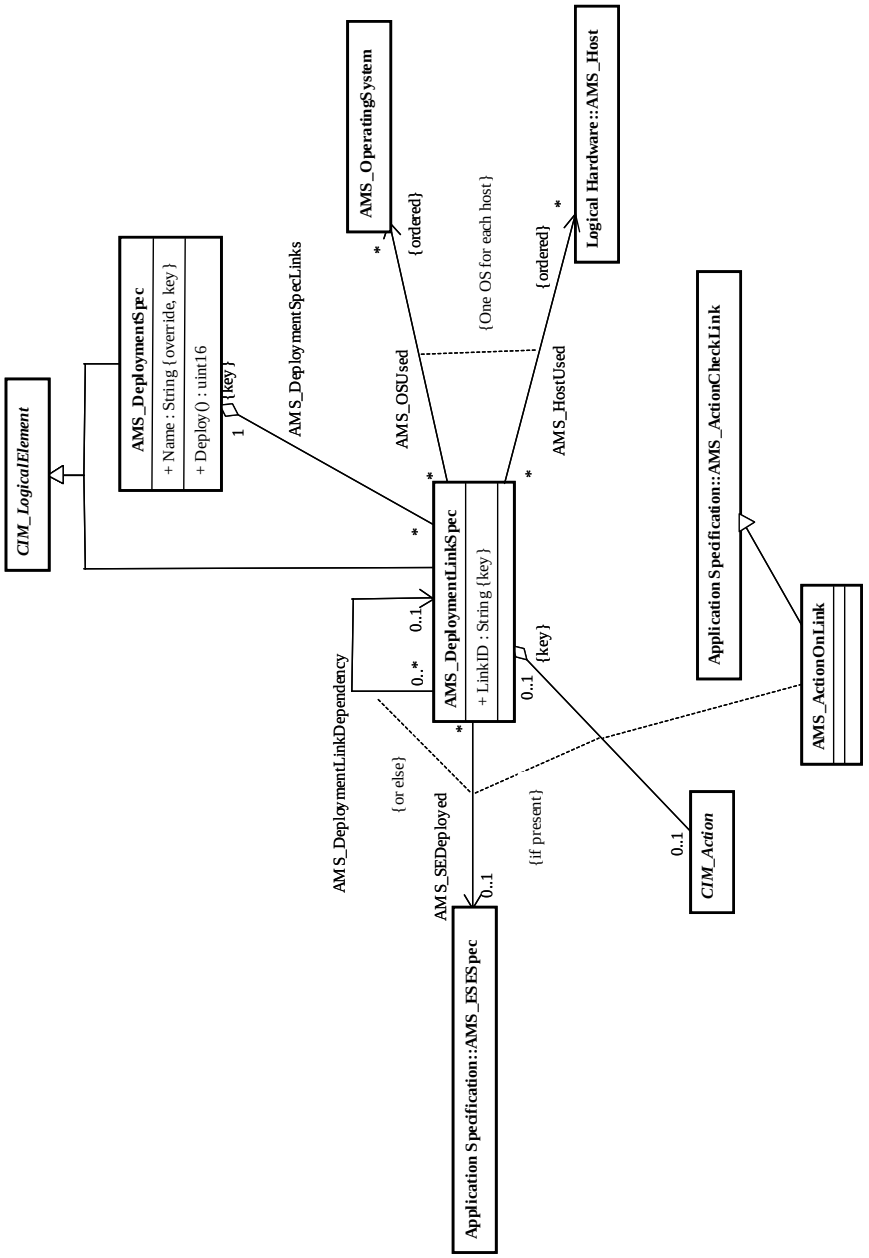


Figure 7.13 - Application Deployment Specification class diagram

The following instantiation diagram shows an example of such elements.

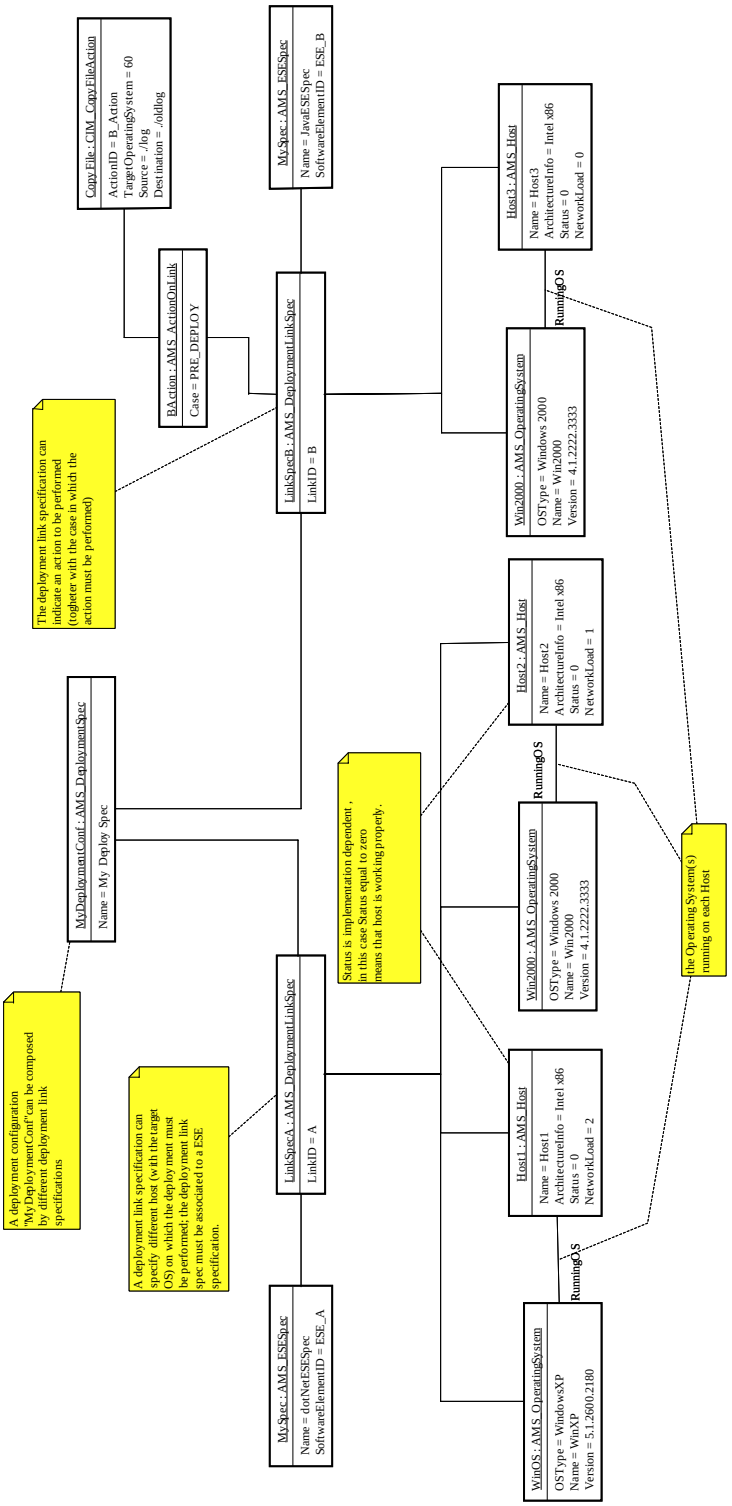


Figure 7.14 - Application Deployment Specification instanciation

7.1.6.1 AMS_ActionOnLink Class

The AMS_ActionOnLink class is an association class between AMS_DeploymentLinkSpec and CIM_Action. It is specifying in which case the action will be used: start, stop, etc.

7.1.6.2 AMS_DeploymentLinkSpec Class

The AMS_DeploymentLinkSpec class models the fact that a software item (application, software system, or software element) will have to run on some hosts. These hosts are defined either as an actual host or with a kind of host (requested hardware).

It belongs to an AMS_DeploymentSpec and is linked to the following:

- either some AMS_Hosts and AMS_OperatingSystem. Each host must correspond with one operating system, and
- either an AMS_ESESpec or another AMS_DeploymentLinkSpec of which the software will be taken into account.

It also may have an action linked with the association class AMS_ActionOnLink, which specifies in which case (start, stop, etc.) this action will have to be used.

Its attribute is a "LinkID" which, with its AMS_DeploymentSpec's global name, takes part of the make-up of its global name.

Attributes	*Type*	
LinkID {key}	String	
Associations	*Multiplicity*	*Class*
AMS_ConfSpecDLS <<optional HSM>>	0 .. 1	AMS_ConfigurationSpecification
AMS_SEDeployed	0 .. 1	AMS_ESESpec
AMS_HostUsed {ordered}	* .. *	AMS_Host
AMS_DeploymentLinkDependency	0 .. 1	AMS_DeploymentLinkSpec
AMS_DeploymentSpecLinks {key}	1 .. 1	AMS_DeploymentSpec
AMS_ActionOnLink	0 .. 1	CIM_Action
AMS_OSUsed {ordered}	* .. *	AMS_OperatingSystem

7.1.6.3 AMS_DeploymentSpec Class

The AMS_DeploymentSpec class models the information needed to define deployment configurations. It gathers AMS_DeploymentLinkSpecs. Its attribute is its name.

Attributes	*Type*	
Name {override, key}	String	
Associations	*Multiplicity*	*Class*

AMS_DeploymentSpecAssoc	0 .. *	AMS_DeploymentConfiguration
AMS_DeploymentSpecLinks	* .. *	AMS_DeploymentLinkSpec
Operations	*Parameters*	*Parameters type*
Deploy	return type	uint16

The following sections review the operation in detail.

7.1.6.3.1 Deploy ( )

This operation deploys the Executable Software Element defined in the links associated with the current deployment specification on the host defined on the same link.

In case of static deployment, all associations with (AMS_Host, etc.) shall be defined beforehand.

In case of dynamic deployment these associations shall be determined at run time. This second case will be the subject of a future extension of the standard. It shall return:

- AMS_RESOURCEERROR if the deployment could not be performed on account of an OS resource starvation error (lack of memory, disk) when performing one of the actions of the associated executable software element specifications.
- AMS_RIGHTERROR if the deployment could not be performed on account of an OS right error when performing one of the actions of the associated executable software element specifications.
- AMS_BADACTION if the deployment could not be performed on account of a badly formed action in the executable software element specification definitions.
- AMS_BADCHECK if the deployment could not be performed on account of a badly formed check in the executable software element specification definitions.
- AMS_NOTCHECKED if at least one of the checks of the associated executable software element specifications was not true while deploying.
- AMS_BADMODELTYPE if the targeted hosts does not support the matching specification model types.
- AMS_DEPLOYFAILED if the element could not start for any other reason.

In all case of error, a full explanation of the cause of the error shall be logged.

After an error, the executable software elements already deployed are in an unknown state.

7.1.7 Application Specification Package

The "Application Specification" package groups the classes needed to model applications so they can be deployed subsequently. This package is a configuration view of applications.

The information needed to manage and monitor applications have been put in the "Application" package.

The main piece here is the specification of an executable software element: an AMS_ESESpec. An executable software element specification is the object which the AMSM service needs to deploy applications (i.e., create an executable software element from its specification) and, subsequently, to manage and monitor it. An actual executable software element will not hold a lot of information in itself since it will use its specification to keep them. Of this application information, the most important are the checks (CIM_Check) and actions (CIM_Action).

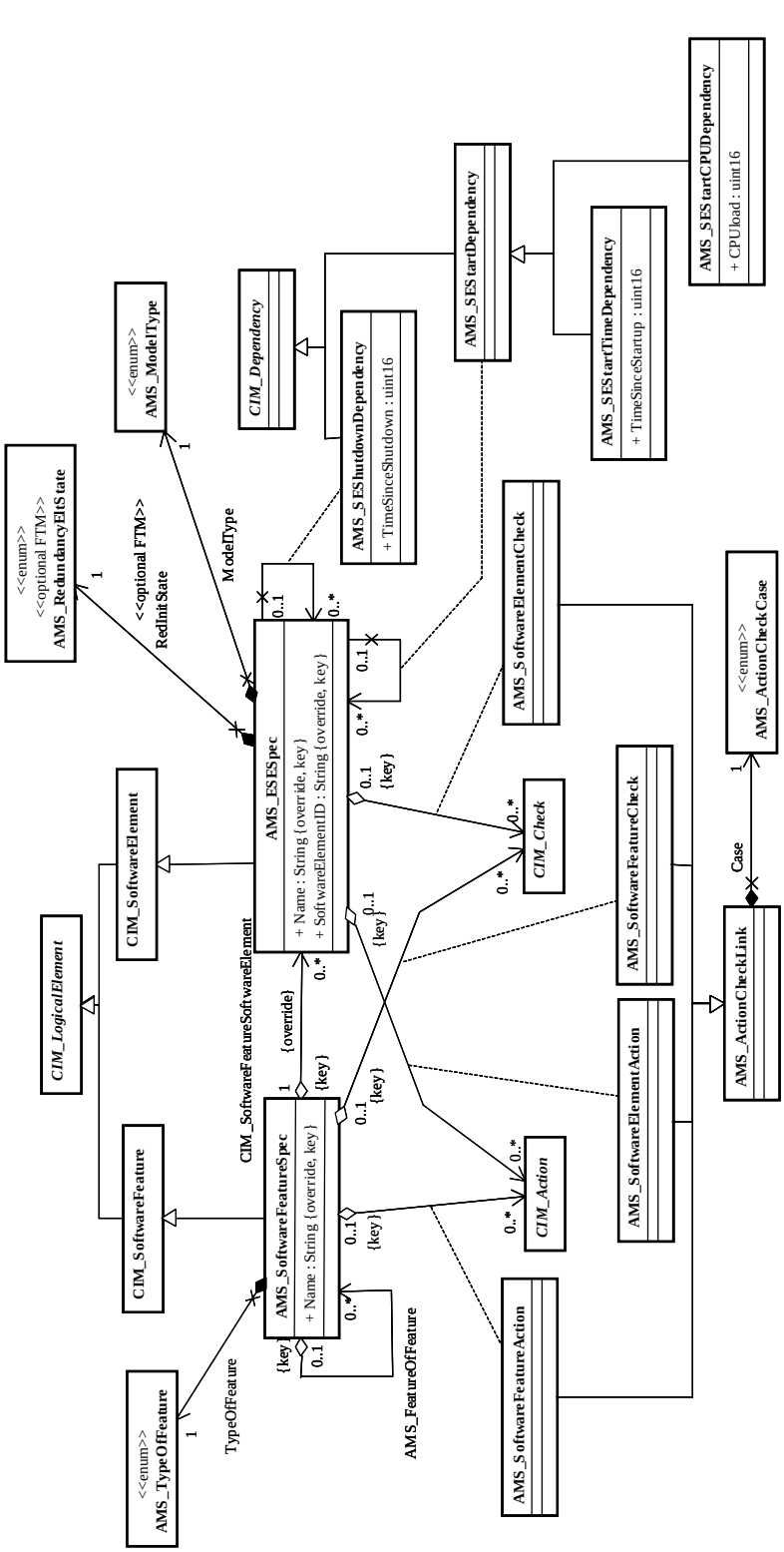


Figure 7.15 - Application Specification class diagram

CIM_Action are "operations that are part of a process to start or shutdown [or deploy] a software element" (from CIM documentation).

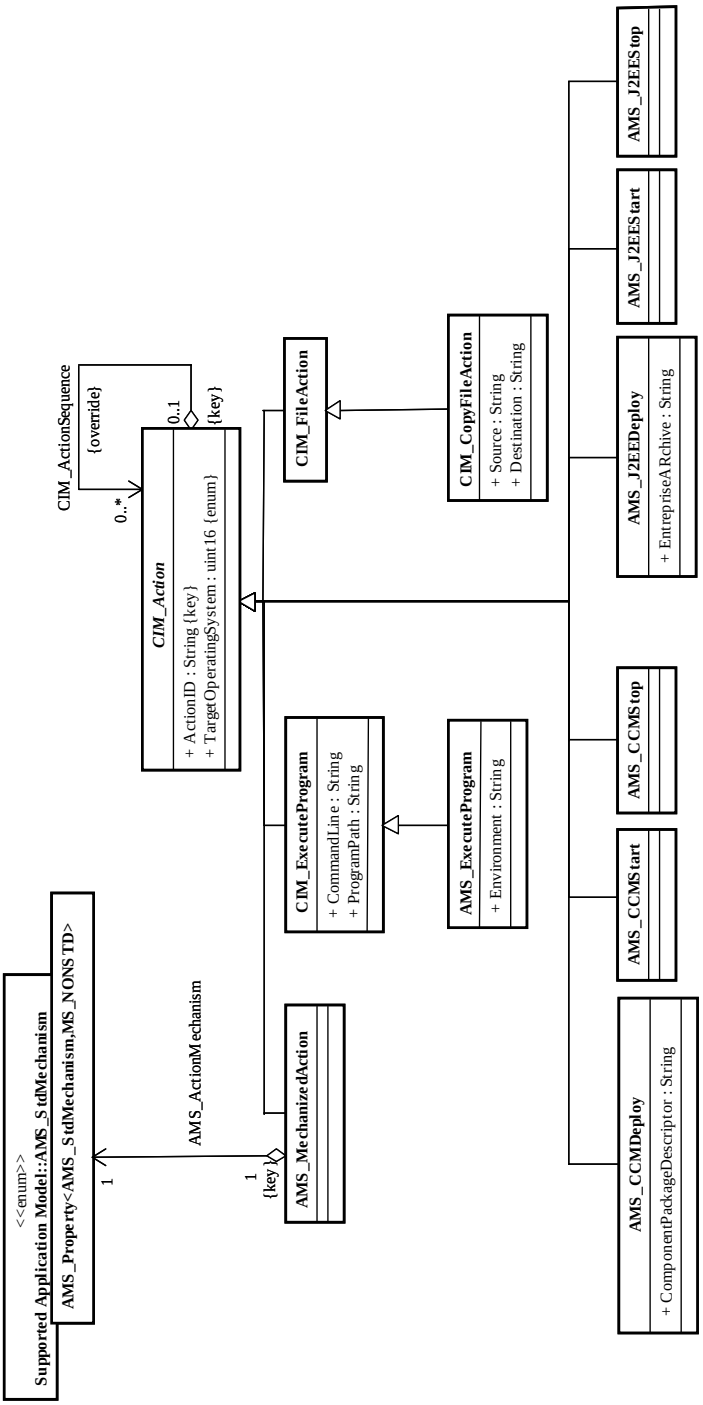


Figure 7.16 - Action Specification class diagram

CIM_Check are "conditions or characteristics that have to be true so as to deploy a Software Element" (from CIM documentation).

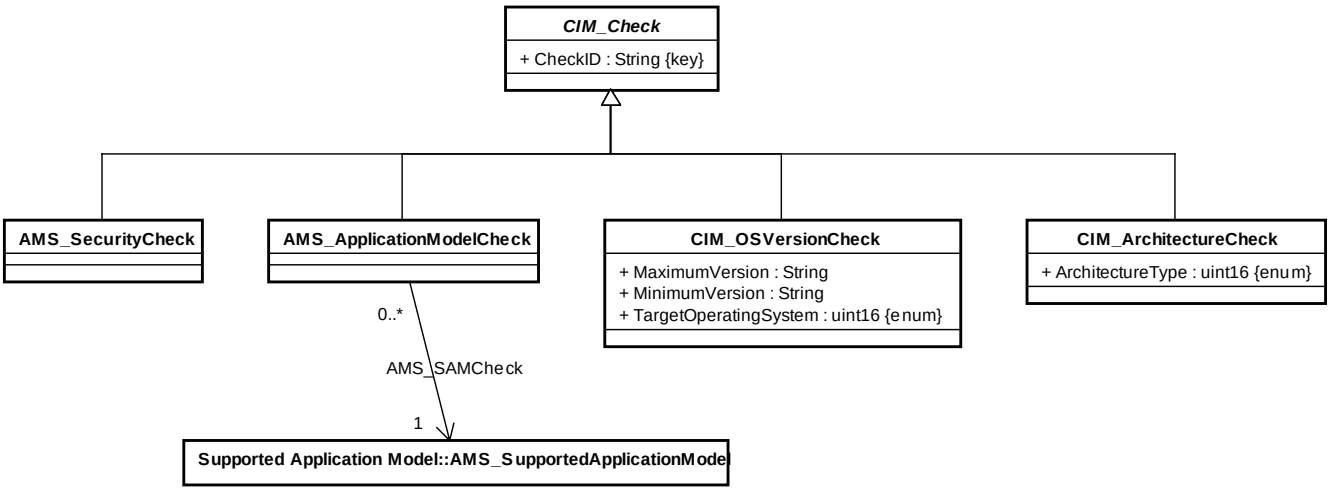


Figure 7.17 - Check Specification class diagram

An association class between AMS_ESESpec and either CIM_Action or CIM_Check specifies the condition (start, stop, deploy) in which the action or check will take place.

Specifications of executable software elements are then gathered in feature specifications, which could be next deployed as systems, applications, load balancing groups, or redundancy groups.

The AMS_ExecuteProgram is a subclass of a CIM_ExecuteProgram. The CIM_ExecuteProgram class contains the Strings CommandLine and ProgramPath. These provide a way of specifying the (OS dependent) command and path, including any arguments required on the command line, to perform a transition of the ESE state diagram. These can be used to start scripts, executables, or container applications such as J2SE applications. The ASM_ExecuteProgram class adds an environment string which can be used to provide an appropriate environment string if required.

A separate instantiation of an AMS_ExecuteProgram object would be created for each supported action given by the AMS_ActionCheckCase enumeration. In an actual implementation of a system, an AMS_ESESpec instance is used to gather the static specification data of a particular Software Element. The AMS_SoftwareElementAction association class is used to associate a particular instance of the AMS_ExecuteProgram object with the AMS_ESESpec for the corresponding Executable Software Element. There will in general be multiple instances of the AMS_SoftwareElementAction association class for each AMS_ESESpec. Each of these instances will have different values for the AMS_ActionCheckCase corresponding to the appropriate action in the ESE state diagram. Each AMS_SoftwareElementAction will associate a particular AMS_ESESpec with a particular instance of an AMS_ExecuteProgram object, which provides the (OS dependent) parameters to take a particular action. Figure 7.17 provides an example of an instance diagram illustrating the use of these classes.

The following instantiation diagram shows an example of application specification objects.

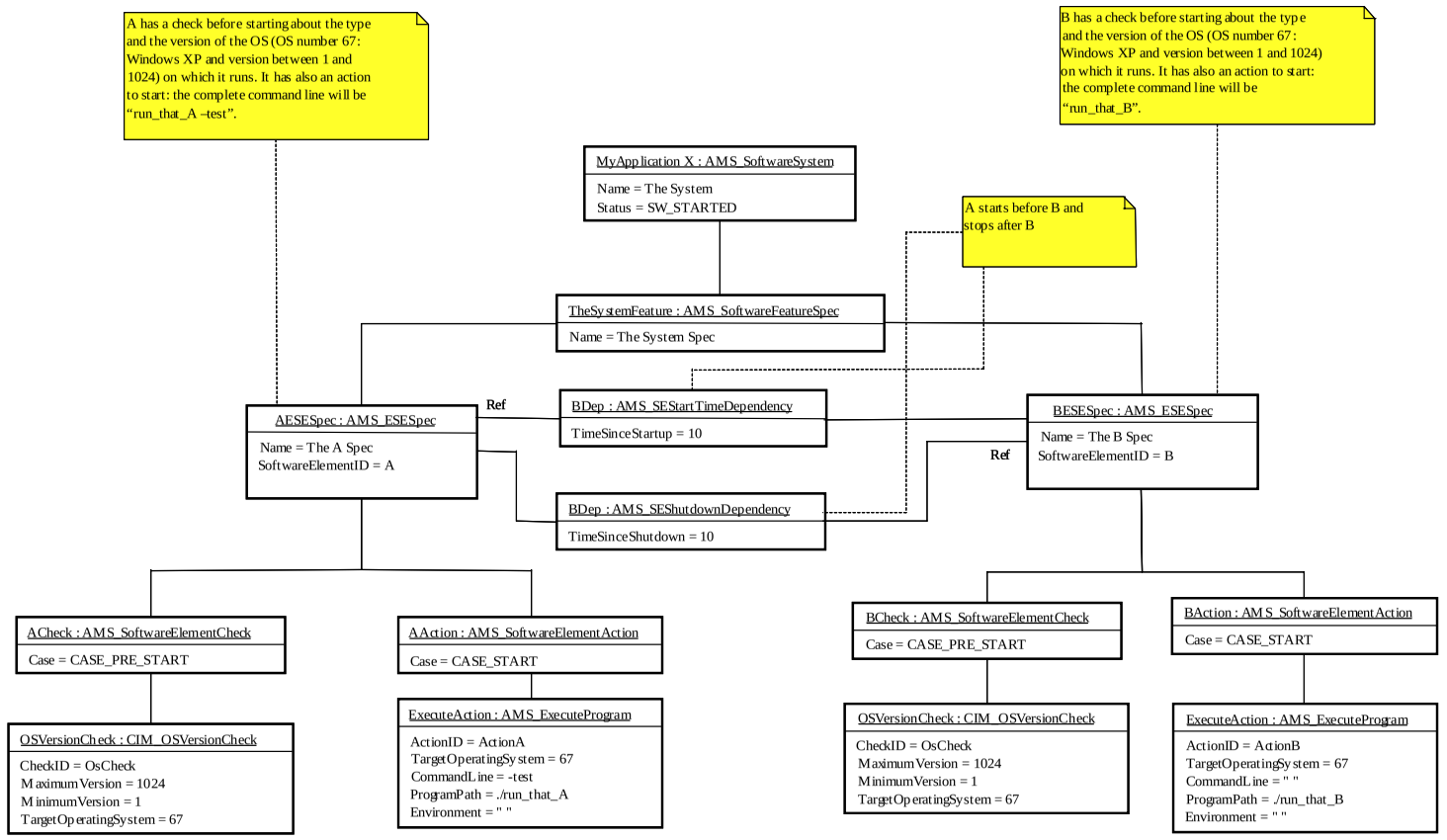


Figure 7.18 - Application specification instantiation

7.1.7.1 AMS_ActionCheckCase Class

The AMS_ActionCheckCase class is the enumeration of all the conditions in which actions or checks could take place:

- CASE_PRE_DEPLOY: just before deployment
- CASE_DEPLOY: during deployment (action: how to deploy, check: is it allowed to deploy)
- CASE_POST_DEPLOY: just after deployment
- CASE_PRE_START: just before a startup
- CASE_START: during startup (action: how to startup, check: is it allowed to startup)
- CASE_POST_START: just after startup
- CASE_PRE_SHUTDOWN: just before shutdown
- CASE_SHUTDOWN: during shutdown (action: how to shutdown, check: is it allowed to shutdown)
- CASE_POST_SHUTDOWN: just after shutdown
- CASE_ALTERNATE_SHUTDOWN: an other way to shutdown the element. This is intended for "last chance" killing (such as "kill -9" on Unix systems).

Enumeration
CASE_PRE_DEPLOY
CASE_DEPLOY
CASE_POST_DEPLOY
CASE_PRE_START
CASE_START
CASE_POST_START
CASE_PRE_SHUTDOWN
CASE_SHUTDOWN
CASE_POST_SHUTDOWN
CASE_ALTERNATE_SHUTDOWN

7.1.7.2 AMS_ActionCheckLink Class

The AMS_ActionCheckLink class is a super-class for all the association classes to CIM_Action and CIM_Check.

It defines the attribute “Case” that contains the condition in which the action or the check will take place.

This condition is taken among the enumeration AMS_ActionCheckCase.

Associations	*Multiplicity*	*Class*
Case	1 .. 1	AMS_ActionCheckCase

7.1.7.3 AMS_ApplicationModelCheck Class

The AMS_ApplicationModelCheck class models the tests in relation to an application model.

One of these checks implies to verify that its application model is supported by the targeted host.

Its attribute provides the AMS_SupportedApplicationModel against witch the targeted host has to be tested.

Associations	*Multiplicity*	*Class*
AMS_SAMCheck	1 .. 1	AMS_SupportedApplicationModel

7.1.7.4 AMS_CCMDeploy Class

The AMS_CCMDeploy causes deployment of Applications and Systems from a component package descriptor (cf [D&C]).

Its attribute is ComponentPackageDescriptor which is the file containing the package descriptor.

Attributes	*Type*	
ComponentPackageDescriptor	String	

7.1.7.5 AMS_CCMStart Class

AMS_CCMStart causes CCM Executable Software Element to be started.

7.1.7.6 AMS_CCMStop Class

AMS_CCMStart causes CCM Executable Software Element to be stopped.

7.1.7.7 AMS_ESESpec Class

The AMS_ESESpec class is designed as a subclass of CIM_SoftwareElement, which is referred in CIM documentation as "a collection of files and associated details...".

It is part of an AMS_SoftwareFeatureSpec and may be associated with CIM_Checks and CIM_Actions.

More specifically, actions and checks are linked to executable software element specifications with an attribute (classes AMS_SoftwareElementAction and AMS_SoftwareElementCheck), which describes in which case the actions or checks must be used: deployment, pre(post)-start, start, pre(post)-shutdown or for an alternate shutdown mechanism.

The AMS_SEShutdownDependency and AMS_SEStartDependency associations with AMS_ESESpecs allow the definition of dependency graphs for, respectively, the shutdown or the start of complex applications. These associations may have the following attributes defined for the links:

- "TimeSinceShutdown" sets a delay before the shutdown of next the software element.
- "TimeSinceStartup" sets a delay before the start of the next software element.
- "CPUload" sets a condition on the CPU before the start of the next software element.

AMS_ESESpec Class attributes and associations are:

- A name and a software id that, with its AMS_SoftwareFeatureSpec's global name, takes part of the make-up of its global name.
- A container type (ModelType).
- An initial redundancy state (primary, slave... - AMS_RedundancyEltState).

Attributes	*Type*	
Name {override, key}	String	
SoftwareElementID {override, key}	String	
Associations	*Multiplicity*	*Class*
CIM_SoftwareFeatureSoftwareElement {key} {override}	1 .. 1	AMS_SoftwareFeatureSpec
ModelType	1 .. 1	AMS_ModelType
AMS_SEStartDependency	0 .. *	AMS_ESESpec
AMS_SoftwareElementCheck	0 .. *	CIM_Check
AMS_SoftwareElementAction	0 .. *	CIM_Action
RedInitState <<optional FTM>>	1 .. 1	AMS_RedundancyEltState
AMS_SEShutdownDependency	0 .. *	AMS_ESESpec

7.1.7.8 AMS_ExecuteProgram Class

The AMS_ExecuteProgram is modeled as a subclass of CIM_ExecuteProgram which is defined has action that "causes programs to be executed. ExecuteProgram can be used to launch the effective software element and/or to launch a JVM and so on." (CIM documentation).

On POSIX systems, launching the program must be done through an exec/fork sequence.

The attributes are:

- CommandLine: "A string that can be executed and invokes program(s), from a system's command line" (CIM documentation).
- ProgramPath: the location or 'path' where the program is run.

- Environment: the set of environment variables to be defined before program can be executed.

Attributes	*Type*	
Environment	String	

7.1.7.9 AMS_J2EEDeploy Class

The AMS_J2EEDeploy causes deployment of Applications and Systems from a J2EE archive (cf [J2EE]).

Its attribute is EntrepriseARchive which is the file containing the archive (.ear).

Attributes	*Type*	
EntrepriseARchive	String	

7.1.7.10 AMS_J2EEStart Class

AMS_J2EEStart causes J2EE Executable Software Element to be started.

7.1.7.11 AMS_J2EEStop Class

AMS_J2EEStart causes J2EE Executable Software Element to be stopped.

7.1.7.12 AMS_MechanizedAction Class

The AMS_MechanizedAction class models the actions that use a mechanism defined by either a normalized enumeration (AMS_StdMechanism) or an implementation-defined string and a value.

For instance, the POSIX mechanism "kill –9" is designed with the AMS_StdMechanism MS_POSIXSIGNAL and the value "9."

Note – Shell scripts have to be launched through the AMS_ExecuteProgram class, which allow to precise the shell used and the script itself in its CommandLine attribute.

Obviously, the mechanism of the AMS_MechanizedAction class will have to match with the mechanisms allowed by the supported application model of the AMS_OperatingSystem on which the action has to be performed. This check will have to be carried out when starting, stopping, or deploying an AMS_ExecutableSoftwareElement.

Associations	*Multiplicity*	*Class*
AMS_ActionMechanism {key}	1	AMS_Property<AMS_StdMechanism,MS_NONSTD>

7.1.7.13 AMS_RedundancyEltState Class

This class belongs to the "Fault Tolerance Management" profile.

The AMS_RedundancyEltState class enumerates possible states of an executable software element regarding its role in a redundancy group:

- REDSTATE_NORG: the element is not in a redundancy group.
- REDSTATE_PRIMARY: the element is the primary in its redundancy group.
- REDSTATE_PASSIVE: the element is one of the passive in its redundancy group.

Enumeration
REDSTATE_NORG
REDSTATE_PRIMARY
REDSTATE_PASSIVE

7.1.7.14 AMS_SecurityCheck Class

The AMS_ApplicationModelCheck class models the tests in relation to the concept of security. The behavior of this class is implementation-dependent.

7.1.7.15 AMS_SEShutdownDependency Class

The AMS_SEShutdownDependency class is an association class among AMS_ESESpec. It defines the shutdown dependency graph with one attribute for each link:

- "TimeSinceShutdown" specifies the delay before shutting down the AMS_ESESpec.

Attributes	*Type*	
TimeSinceShutdown	uint16	

7.1.7.16 AMS_SEStartCPUDependency Class

The AMS_SEStartDependency class is an association class among AMS_ESESpec. It defines the start CPU load dependency graph with one attribute for each links:

- "CPUload" specifies maximum value of CPU load present on the host before the executable software element is to be started.

Attributes	*Type*	
CPUload	uint16	

7.1.7.17 AMS_SEStartDependency Class

The AMS_SEStartDependency class is an association class among AMS_ESESpec. It defines the start dependency graph having two subclasses: on the one hand for time dependency, and on the other hand for CPU load dependency.

7.1.7.18 AMS_SEStartTimeDependency Class

The AMS_SEStartDependency class is an association class among AMS_ESESpec. It defines the start time dependency graph with one attribute for each links:

- "TimeSinceStartup" specifies the delay before starting the AMS_ESESpec.

Attributes	*Type*	
TimeSinceStartup	uint16	

7.1.7.19 AMS_SoftwareElementAction Class

The AMS_SoftwareElementAction class is an association class between AMS_ESESpec and CIM_Action. It specifies in which case the action will be used: start, stop, etc.

7.1.7.20 AMS_SoftwareElementCheck Class

The AMS_SoftwareElementCheck class is an association class between AMS_SoftwareFeature and CIM_Check. It specifies in which case the check will be used: start, stop, etc.

7.1.7.21 AMS_SoftwareFeatureAction Class

The AMS_SoftwareFeatureAction class is an association class between AMS_SoftwareFeatureSpec and CIM_Action. It specifies in which case the action will be used: start, stop, etc.

7.1.7.22 AMS_SoftwareFeatureCheck Class

The AMS_SoftwareFeatureCheck class is an association class between AMS_SoftwareFeature and CIM_Check. It specifies in which case the check will be used: start, stop.

7.1.7.23 AMS_SoftwareFeatureSpec Class

The AMS_SoftwareFeatureSpec class is designed as a subclass of CIM_SoftwareFeature, which is referred as "a concept [that] allows software products or application systems to be decomposed into units that have a meaning to users rather than units that reflect how the product or application was built (i.e., software elements" in CIM documentation).

It may be part of another AMS_SoftwareFeatureSpec in a "system of system" spirit.

It gathers AMS_ESESpecs or other AMS_SoftwareFeatureSpecs and has CIM_Checks and CIM_Actions.

More specifically, actions and checks are linked to executable software element specifications with an attribute (classes AMS_SoftwareFeatureAction and AMS_SoftwareFeatureCheck), which describes in which case the actions or checks must be used: deployment, pre(post)-start, start, pre(post)-shutdown or for an alternate shutdown mechanism.

The attributes or associations are:

- A name that, with the global name of the AMS_SoftwareFeatureSpec to which it belongs, takes part of the make-up of its global name.
- A type of feature (when deployed): system, application, redundancy group, or load balancing group.

Attributes	*Type*	
Name {override, key}	String	
Associations	*Multiplicity*	*Class*

AMS_FeatureOfFeature {key}	0 .. 1	AMS_SoftwareFeatureSpec
AMS_FeatureOfFeature	0 .. *	AMS_SoftwareFeatureSpec
CIM_SoftwareFeatureSoftwareElement {override}	0 .. *	AMS_ESESpec
TypeOfFeature	1 .. 1	AMS_TypeOfFeature
AMS_SoftwareFeatureCheck	0 .. *	CIM_Check
AMS_SoftwareFeatureAction	0 .. *	CIM_Action

7.1.7.24 AMS_TypeOfFeature Class

The AMS_TypeOfFeature class enumerates the kinds of objects in which an AMS_SoftwareFeatureSpec can be deployed.

Enumeration
SYSTEM
APPLICATION
REDUNDANCY_GROUP
LOADBALANCING_GROUP

7.1.8 CIM Package

The CIM package is a collection of CIM classes used by the AMSM service. The classes are a subset of classes defined by the CIM standard. The documentation shown afterward is an extract from the CIM documentation.

7.1.8.1 CIM_Action Class

A CIM_Action is an operation that is part of a process to either create a CIM_SoftwareElement or to eliminate the CIM_SoftwareElement. The attributes are:

- "ActionID:" The ActionID property is a unique identifier assigned to a particular Action for a SoftwareElement. (key).

Issue 11519 - Vista

- "TargetOperatingSystem:" The Target Operating System of the SoftwareElement being acted upon. The enumeration is: Unknown, Other, MACOS, ATTUNIX, DGUX, DECNT, Tru64 UNIX, OpenVMS, HPUX, AIX, MVS, OS400, OS/2, JavaVM, MSDOS, WIN3x, WIN95, WIN98, WINNT, WINCE, NCR3000, NetWare, OSF, DC/OS, Reliant UNIX, SCO UnixWare, SCO OpenServer, Sequent, IRIX, Solaris, SunOS, U6000, ASERIES, TandemNSK, TandemNT, BS2000, LINUX, Lynx, XENIX, VM, Interactive UNIX, BSDUNIX, FreeBSD, NetBSD, GNU Hurd, OS9, MACH Kernel, Inferno, QNX, EPOC, IxWorks, VxWorks, MiNT, BeOS, HP MPE, NextStep, PalmPilot, Rhapsody, Windows 2000, Dedicated, OS/390, VSE, TPF, Windows (R) Me, Caldera Open UNIX, OpenBSD, Not Applicable, Windows XP, z/OS, Windows Vista.

Attributes	*Type*	

ActionID {key}	String	
TargetOperatingSystem {enum}	uint16	
Associations	*Multiplicity*	*Class*
CIM_ActionSequence {key} {override}	0 .. 1	CIM_Action
CIM_ActionSequence {override}	0 .. *	CIM_Action
AMS_SoftwareElementAction {key}	0 .. 1	AMS_ESESpec
AMS_SoftwareFeatureAction {key}	0 .. 1	AMS_SoftwareFeatureSpec
AMS_ActionOnLink {key}	0 .. 1	AMS_DeploymentLinkSpec

7.1.8.2 CIM_AdminDomain Class

This class belongs to the "Hardware System Management" profile.

This is a special grouping of CIM_ManagedSystemElements. The grouping is viewed as a single entity, reflecting that all of its components are administered similarly, either by the same user, group of users, or policy. It serves as an aggregation point to associate one or more of the following elements: network devices, such as routers and switches, servers, and other resources that can be accessed by end systems. This grouping of devices plays an essential role in ensuring that the same administrative policy and actions are applied to all of the devices in the grouping. The specific behavior and/or semantics of the AdminDomain can be identified through its aggregated and associated entities.

The attributes are:

- "Name:" The inherited Name serves as key of a System instance in an enterprise environment. (key).

Attributes	*Type*	
Name {key}	String	

7.1.8.3 CIM_ApplicationSystem Class

The CIM_ApplicationSystem class represents an application or a software system that supports a particular business function and that can be managed as an independent unit.

7.1.8.4 CIM_ArchitectureCheck Class

ArchitectureCheck specifies the hardware platform on which a SoftwareElement can run. The processors on the relevant computer system do not need to satisfy the Check.

The attributes are:

- "ArchitectureType:" The ArchitectureType property identifies a particular type of architecture or architectural family that is required to properly execute a particular SoftwareElement. The intent is to capture the details about the machine instructions exploited by the executables of the SoftwareElement. The enumeration is: Other, Unknown, 8086, 80286, 80386, 80486, 8087, 80287, 80387, 80487, Pentium(R) brand, Pentium(R) Pro, Pentium(R) II, Pentium(R) processor with MMX(TM) technology, Celeron(TM), Pentium(R) II Xeon(TM), Pentium(R) III, M1 Family, M2 Family, K5

Family, K6 Family, K6-2, K6-3, AMD Athlon(TM) Processor Family, AMD(R) Duron(TM) Processor, AMD29000 Family, K6-2%2B, Power PC Family, Power PC 601, Power PC 603, Power PC 603%2B, Power PC 604, Power PC 620, Power PC X704, Power PC 750, Alpha Family, Alpha 21064, Alpha 21066, Alpha 21164, Alpha 21164PC, Alpha 21164a, Alpha 21264, Alpha 21364, MIPS Family, MIPS R4000, MIPS R4200, MIPS R4400, MIPS R4600, MIPS R10000, SPARC Family, SuperSPARC, microSPARC II, microSPARC IIep, UltraSPARC, UltraSPARC II, UltraSPARC IIi, UltraSPARC III, UltraSPARC IIIi, 68040, 68xxx Family, 68000, 68010, 68020, 68030, Hobbit Family, Crusoe(TM) TM5000 Family, Crusoe(TM) TM3000 Family, Weitek, Itanium(TM) Processor, AMD Athlon(TM) 64 Processor Family, AMD Opteron(TM) Processor Family, PA-RISC Family, PA-RISC 8500, PA-RISC 8000, PA-RISC 7300LC, PA-RISC 7200, PA-RISC 7100LC, PA-RISC 7100, V30 Family, Pentium(R) III Xeon(TM), Pentium(R) III Processor with Intel(R) SpeedStep(TM) Technology, Pentium(R) 4, Intel(R) Xeon(TM), AS400 Family, Intel(R) Xeon(TM) processor MP, AMD Athlon(TM) XP Family, AMD Athlon(TM) MP Family, Intel(R) Itanium(R) 2, Intel(R) Pentium(R) M processor, K7, S/390 and zSeries Family, ESA/390 G4, ESA/390 G5, ESA/390 G6, z/Architectur base, i860, i960, SH-3, SH-4, ARM, StrongARM, 6x86, MediaGX, MII, WinChip, DSP, Video Processor.

Attributes	*Type*
ArchitectureType {enum}	uint16

7.1.8.5 CIM_Check Class

A CIM_Check is a condition or characteristic that is expected to be true in an environment defined or scoped by an instance of a CIM_ComputerSystem. The attributes are:

- "CheckID": An identifier used to uniquely identify the Check. (key).

Attributes	*Type*	
CheckID {key}	String	
Associations	*Multiplicity*	*Class*
AMS_SoftwareElementCheck {key}	0 .. 1	AMS_ESESpec
AMS_SoftwareFeatureCheck {key}	0 .. 1	AMS_SoftwareFeatureSpec

7.1.8.6 CIM_ComputerSystem Class

A class derived from System that is a special collection of ManagedSystemElements. This collection provides compute capabilities and serves as aggregation point to associate one or more of the following elements: FileSystem, OperatingSystem, Processor, and Memory (Volatile and/or NonVolatile Storage).

7.1.8.7 CIM_ConnectivityCollection Class

This class belongs to the "Hardware System Management" profile.

A ConnectivityCollection groups together a set of ProtocolEndpoints of the same 'type' (i.e., class) which are able to communicate with each other.

Its attributes give:

- "InstanceID:" Within the scope of the instantiating Namespace, InstanceID opaquely and uniquely identifies an instance of this class. In order to ensure uniqueness within the NameSpace, the value of InstanceID SHOULD be

constructed using the following 'preferred' algorithm: <OrgID>:<LocalID> Where <OrgID> and <LocalID> are separated by a colon ':', and where <OrgID> MUST include a copyrighted, trademarked or otherwise unique name that is owned by the business entity creating/defining the InstanceID, or is a registered ID that is assigned to the business entity by a recognized global authority (This is similar to the <Schema Name>_<Class Name> structure of Schema class names.) In addition, to ensure uniqueness <OrgID> MUST NOT contain a colon (':'). When using this algorithm, the first colon to appear in InstanceID MUST appear between <OrgID> and <LocalID>. <LocalID> is chosen by the business entity and SHOULD not be re-used to identify different underlying (real-world) elements. If the above 'preferred' algorithm is not used, the defining entity MUST assure that the resultant InstanceID is not re-used across any InstanceIDs produced by this or other providers for this instance's NameSpace. For DMTF defined instances, the 'preferred' algorithm MUST be used with the <OrgID> set to 'CIM'.

Attributes	*Type*	
InstanceID {key}	uint64	
Associations	*Multiplicity*	*Class*
CIM_MemberOfCollection	* .. *	CIM_ProtocolEndPoint
CIM_HostedCollection {key}	1 .. 1	CIM_Network

7.1.8.8 CIM_CopyFileAction Class

CIM_CopyFileAction specifies the files to be moved or copied to a new location. The to/from information for the copy is specified using either the ToDirectorySpecification/ FromDirectorySpecification or the ToDirectoryAction/ FromDirectoryAction associations. The first set is used when the source and/or the target are to exist before any Actions are taken. The second set is used when the source and/or target are created as a part of a previous Action (specified using the association, CIM_ActionSequence).

Attributes	*Type*	
Source	String	
Destination	String	
Associations	*Multiplicity*	*Class*
CIM_ToDirectoryAction	1 .. 1	CIM_DirectorySpecification
CIM_FromDirectorySpecification	1 .. 1	CIM_DirectorySpecification

7.1.8.9 CIM_Dependency Class

CIM_Dependency is a generic association used to establish dependency relationships between ManagedElements.

7.1.8.10 CIM_DirectorySpecification Class

The CIM_DirectorySpecification class captures the major directory structure of a SoftwareElement. This class is used to organize the files of a SoftwareElement into manageable units that can be relocated on a computer system. Its attributes give:

- "DirectoryPath:" The DirectoryPath property is used to capture the name of a directory. The value supplied by an application provider is actually a default or recommended path name. The value can be changed for a particular environment.

Attributes	*Type*	
DirectoryPath	String	
Associations	*Multiplicity*	*Class*
CIM_ToDirectoryAction {key}	0 .. 1	CIM_CopyFileAction
CIM_FromDirectorySpecification {key}	0 .. 1	CIM_CopyFileAction

7.1.8.11 CIM_Display Class

This class belongs to the "Hardware System Management" profile.

Display is a superclass for grouping the miscellaneous display devices that exist.

7.1.8.12 CIM_EnabledLogicalElement Class

This class extends CIM_LogicalElement to abstract the concept of an element that is enabled and disabled, such as a CIM_LogicalDevice or a CIM_ServiceAccessPoint.

7.1.8.13 CIM_EthernetPort Class

This class belongs to the "Hardware System Management" profile.

Capabilities and management of an EthernetPort.

7.1.8.14 CIM_EthernetPortStatistics Class

This class belongs to the "Hardware System Management" profile. The EthernetPortStatistics class describes the statistics for the EthernetPort. Its attributes give:

- "InstanceID:" Within the scope of the instantiating Namespace, InstanceID opaquely and uniquely identifies an instance of this class. In order to ensure uniqueness within the NameSpace, the value of InstanceID SHOULD be constructed using the following 'preferred' algorithm: <OrgID>:<LocalID> Where <OrgID> and <LocalID> are separated by a colon ':', and where <OrgID> MUST include a copyrighted, trademarked or otherwise unique name that is owned by the business entity creating/defining the InstanceID, or is a registered ID that is assigned to the business entity by a recognized global authority (This is similar to the <Schema Name>_<Class Name> structure of Schema class names.) In addition, to ensure uniqueness <OrgID> MUST NOT contain a colon (':'). When using this algorithm, the first colon to appear in InstanceID MUST appear between <OrgID> and <LocalID>. <LocalID> is chosen by the business entity and SHOULD not be re-used to identify different underlying (real-world) elements. If the above 'preferred' algorithm is not used, the defining entity MUST assure that the resultant InstanceID is not re-used across any InstanceIDs produced by this or other providers for this instance's NameSpace. For DMTF defined instances, the 'preferred' algorithm MUST be used with the <OrgID> set to 'CIM'.
- "BytesTransmitted:" The total number of bytes transmitted, including framing characters.

- "BytesReceived:" The total number of bytes received, including framing characters.
- "PacketsTransmitted:" The total number of packets transmitted.
- "PacketsReceived:" The total number of packets received.

Attributes	*Type*	
InstanceID {key}	uint64	
BytesTransmitted	uint64	
BytesReceived	uint64	
PacketsTransmitted	uint64	
PacketsReceived	uint64	
Associations	*Multiplicity*	*Class*
CIM_ElementStatisticalData	1 .. 1	CIM_LANEndPoint

7.1.8.15 CIM_ExecuteProgram Class

ExecuteProgram causes programs to be executed on the computer system that defines the Action's environment. Its attribute give:

- "CommandLine:" A string that can be executed and invokes program(s), from a system’s command line.
- "ProgramPath:" The location or 'path' where the program is found.

Attributes	*Type*	
CommandLine	String	
ProgramPath	String	

7.1.8.16 CIM_FileAction Class

FileAction locates files that already exist on the CIM_ComputerSystem that defines the Action’s environment. These files are removed or moved/copied to a new location.

7.1.8.17 CIM_IPProtocolEndPoint Class

This class belongs to the "Hardware System Management" profile. A ProtocolEndpoint that is dedicated to running IP. Its attributes give:

- "IPv4Address:" The IPv4 address that this ProtocolEndpoint represents.
- "IPv6Address:" The IPv6 address that this ProtocolEndpoint represents.
- "SubnetMask:" The mask for the IPv4 address of this ProtocolEndpoint, if one is defined.

- "PrefixLenght:" The prefix length for the IPv6 address of this Protocol Endpoint, if one is defined.

Attributes	*Type*
IPv4Address	String
IPv6Address	String
SubnetMask	String
PrefixLength	uint8

7.1.8.18 CIM_Indication Class

CIM_Indication is the abstract root class for all notifications about changes in schema, objects and their data, and about events detected by providers and instrumentation. Subclasses represent specific types of notifications.

Its attribute give:

- "IndicationTime:" The time and date of creation of the Indication. The property may be set to NULL if the entity creating the Indication is not capable of determining this information. Note that IndicationTime may be the same for two Indications that are generated in rapid succession.

Attributes	*Type*
IndicationTime	datetime

7.1.8.19 CIM_LANEndPoint Class

This class belongs to the "Hardware System Management" profile. A communication endpoint which, when its associated interface device is connected to a LAN, may send and receive data frames. LANEndpoints include Ethernet, Token Ring, and FDDI interfaces. Its attributes give:

- "LANID:" A label or identifier for the LAN Segment to which the Endpoint is connected. If the Endpoint is not currently active/connected or this information is not known, then LANID is NULL.
- "MACAddress:" The principal unicast address used in communication with the CIM_LANEndpoint. The MAC address is formatted as twelve hexadecimal digits (e.g., "010203040506"), with each pair representing one of the six octets of the MAC address in "canonical" bit order according to RFC 2469.
- "AliasAddresses:" Other unicast addresses that may be used to communicate with the CIM_LANEndpoint.
- "GroupAddresses:" Multicast addresses to which the CIM_LANEndpoint listens.
- "MaxDataSize:" The largest information field that may be sent or received by the LANEndpoint.

The CIM_BindsToLANEndPoint association makes explicit the dependency of a CIM_ProtocolEndpoint on an underlying CIM_LANEndpoint, on the same system.

Attributes	*Type*	
LANID	String	
MACAddress	String	

AliasAddresses	String	
GroupAddresses	String	
MaxDataSize	uint32	
Associations	*Multiplicity*	*Class*
CIM_ElementStatisticalData	* .. *	CIM_EthernetPortStatistics
CIM_BindsToLANEndPoint	* .. *	CIM_ServiceAccessPoint

7.1.8.20 CIM_Location Class

This class belongs to the "Hardware System Management" profile. The Location class specifies the position and address of a PhysicalElement. Its attribute give:

- "Name:" Name is a free-form string defining a label for the Location. It is a part of the key for the object.

Attributes	*Type*	
Name {key}	String	
Associations	*Multiplicity*	*Class*
CIM_PhysicalElementLocation	0 .. *	CIM_PhysicalElement
CIM_ElementLocation {override}	0 .. *	AMS_ComputerSystem

7.1.8.21 CIM_Log Class

Log represents any type of event, error or informational register or chronicle. The object describes the existence of the log and its characteristics. Log does not dictate the form of the data represented or how records/messages are stored in the log and/or accessed. Subclasses will define the appropriate methods and behavior. Its attributes give:

- "MaxNumberOfRecords:" Maximum number of records that can be captured in the Log. If undefined, a value of zero should be specified.
- "CurrentNumberOfRecord:" Current number of records in the Log.
- "Name:" The Name property defines the label by which the object is known.

Attributes	*Type*	
MaxNumberOfRecords	uint64	
CurrentNumberOfRecord	uint64	
Name	String	
Operations	*Parameters*	*Parameters type*
ClearLog	return type	uint32

The following sections review the operation in detail.

7.1.8.21.1 ClearLog ( )

Purge the log storage area.

7.1.8.22 CIM_LogicalDevice Class

This class belongs to the "Hardware System Management" profile.

An abstraction or emulation of a hardware entity, that may or may not be Realized in physical hardware. Any characteristics of a CIM_LogicalDevice that are used to manage its operation or configuration are contained in, or associated with, the CIM_LogicalDevice object. Various configurations could exist for a CIM_LogicalDevice. These configurations could be contained in Setting objects and associated with the LogicalDevice. Its attribute gives:

- "DevideID:" An address or other identifying information to uniquely name the LogicalDevice (key).

Attributes	*Type*	
DeviceID {key}	String	
Associations	*Multiplicity*	*Class*
CIM_Realizes	* .. *	CIM_PhysicalElement
CIM_SystemComponent {key} {override}	0 .. 1	AMS_ComputerSystem

7.1.8.23 CIM_LogicalDisk Class

This class belongs to the "Hardware System Management" profile.

A CIM_LogicalDisk is a presentation of a contiguous range of logical blocks that is identifiable by a FileSystem via the Disk's DeviceId (key) field. For example in a Windows environment, the DeviceID field would contain a drive letter. In a Unix environment, it would contain the access path; and in a NetWare environment, DeviceID would contain the volume name. LogicalDisks are typically built on a DiskPartition or Storage Volume (for example, exposed by a software volume manager) using the LogicalDiskBasedOnPartition or LogicalDiskBasedOn Volume associations.

7.1.8.24 CIM_LogicalElement Class

CIM_LogicalElement is a base class for all the components of a System that represent abstract system components, such as Files, Processes, or LogicalDevices.

7.1.8.25 CIM_LogicalPort Class

This class belongs to the "Hardware System Management" profile.

The abstraction of a port or connection point of a Device. This object should be instantiated when the Port has independent management characteristics from the Device that includes it. Examples are a Fibre Channel Port and a USB Port.

7.1.8.26 CIM_ManagedElement Class

CIM_ManagedElement is an abstract class that provides a common superclass (or top of the inheritance tree) for the non-association classes in the CIM Schema.

7.1.8.27 CIM_Memory Class

This class belongs to the "Hardware System Management" profile.

Capabilities and management of Memory-related CIM_LogicalDevices.

7.1.8.28 CIM_Network Class

This class belongs to the "Hardware System Management" profile.

CIM_Network is a subclass of CIM_AdminDomain that groups interconnected networking and computing objects capable of exchanging information. Instances of CIM_Network can represent an enterprise's global network or specific connectivity domains within the global network. These concepts are similar to those defined for the Network object in ITU's M.3100 specification.

Associations	*Multiplicity*	*Class*
CIM_HostedCollection	* .. *	CIM_ConnectivityCollection
AMS_SubnetComponent	* .. *	CIM_Network

7.1.8.29 CIM_NetworkPort Class

This class belongs to the "Hardware System Management" profile.

CIM_NetworkPort is the logical representation of network communications hardware - a physical connector and the setup/operation of the network chips, at the lowest layers of a network stack.

7.1.8.30 CIM_NextHopIPRoute Class

This class belongs to the "Hardware System Management" profile.

CIM_NextHopIPRoute specifies routing in an IP network.

The attributes are:

- "RouteDerivation:" An enumerated integer indicating how the route was derived. This is useful for display and query purposes. Enumeration is: Unknown, Other, Connected, User-Defined, IGRP, EIGRP, RIP, Hello, EGP, BGP, ISIS, OSPF.
- "DestinationMask:" The mask for the Ipv4 destination address.
- "PrefixLength:" The prefix length for the IPv6 destination address.
- "AddressType:" An enumeration that describes the format of the address properties. The enumeration is: Unknown, IPv4, IPv6.

Attributes	*Type*
RouteDerivation	uint16
DestinationMask	String
PrefixLength	uint8
AddressType	uint16

7.1.8.31 CIM_NextHopRoute Class

This class belongs to the "Hardware System Management" profile.

CIM_NextHopRoute represents one of a series of 'hops' to reach a network destination. A route is administratively defined, or calculated/learned by a particular routing process. The attributes are:

- "InstanceID:" Within the scope of the instantiating Namespace, InstanceID opaquely and uniquely identifies an instance of this class. In order to ensure uniqueness within the NameSpace, the value of InstanceID SHOULD be constructed using the following 'preferred' algorithm: <OrgID>:<LocalID> Where <OrgID> and <LocalID> are separated by a colon ':', and where <OrgID> MUST include a copyrighted, trademarked or otherwise unique name that is owned by the business entity creating/defining the InstanceID, or is a registered ID that is assigned to the business entity by a recognized global authority. (This is similar to the <Schema Name>_<Class Name> structure of Schema class names.) In addition, to ensure uniqueness <OrgID> MUST NOT contain a colon (':'). When using this algorithm, the first colon to appear in InstanceID MUST appear between <OrgID> and <LocalID>. <LocalID> is chosen by the business entity and SHOULD not be re-used to identify different underlying (real-world) elements. If the above 'preferred' algorithm is not used, the defining entity MUST assure that the resultant InstanceID is not re-used across any InstanceIDs produced by this or other providers for this instance's NameSpace. For DMTF defined instances, the 'preferred' algorithm MUST be used with the <OrgID> set to 'CIM'. (key).
- "DestinationAddress:" The address which serves as the destination to be reached.
- "AdminDistance:" The specific administrative distance of this route, overriding any default distances specified by the system or routing service.
- "RouteMetric:" RouteMetric provides a numeric indication as to the preference of this route, compared to other routes that reach the same destination.
- "IsStatic:" TRUE indicates that this is a static route, and FALSE indicates a dynamically-learned route.
- "TypeOfRoute:" An enumerated integer indicating whether the route is administrator-defined (value%3D2), computed (via a routing protocol/algorithm, value%3D3) or the actual route implemented in the network (value%3D4). The default is a computed route. The enumeration is: Administrator Defined Route (2), Computed Route (3), Actual Route (4).

The RouteUsesEndpoint association depicts the relationship between a next hop route and the local Endpoint that is used to transmit the traffic to the 'next hop.'

The AssociatedNextHop association depicts the relationship between a route and the specification of its next hop. The next hop is external to a System, and hence is defined as a kind of CIM_RemoteServiceAccessPoint. Note that this relationship is independent of CIM_RouteUsesEndpoint (the local Endpoint used to transmit the traffic), and both may be defined for a route.

Attributes	*Type*	
InstanceID {key}	String	
DestinationAddress	String	
AdminDistance	uint16	
RouteMetric	uint16	
IsStatic	boolean	
TypeOfRoute {enum}	uint16	
Associations	*Multiplicity*	*Class*
CIM_AssociatedNextHop	0 .. 1	CIM_RemoteServiceAccessPoint
CIM_RouteUsesEndpoint	0 .. 1	CIM_ProtocolEndPoint
CIM_HostedRoute {key} {override}	0 .. 1	AMS_Host
CIM_HostedRoute {key} {override}	0 .. 1	AMS_Router

7.1.8.32 CIM_OSVersionCheck Class

The OSVersionCheck class specifies the versions of the operating system that can support/execute this Software Element.

This Check can be for a specific, minimum, maximum or a range of releases of an OS. To identify a specific version of the OS, the minimum and maximum versions must be the same. To specify a minimum, only the minimum version needs to be defined. To specify a maximum version, only the maximum version needs to be defined. To specify a range, both minimum and maximum versions need to be defined.

The attributes are:

- "MaximumVersion:" Maximum version of the required operating system. The value is encoded as <major>.<minor>.<revision> or <major>.<minor><letter revision>.
- "MinimumVersion:" Minimum version of the required operating system. The value is encoded as <major>.<minor>.<revision> or <major>.<minor><letter revision>.

Issue 11519 - Vista

- "TargetOperatingSystem:" The Target Operating System of the SoftwareElement being checked. The enumeration is: Unknown, Other, MACOS, ATTUNIX, DGUX, DECNT, Tru64 UNIX, OpenVMS, HPUX, AIX, MVS, OS400, OS/2, JavaVM, MSDOS, WIN3x, WIN95, WIN98, WINNT, WINCE, NCR3000, NetWare, OSF, DC/OS, Reliant UNIX, SCO UnixWare, SCO OpenServer, Sequent, IRIX, Solaris, SunOS, U6000, ASERIES, TandemNSK, TandemNT, BS2000, LINUX, Lynx, XENIX, VM, Interactive UNIX, BSDUNIX, FreeBSD, NetBSD, GNU Hurd, OS9, MACH

Kernel, Inferno, QNX, EPOC, IxWorks, VxWorks, MiNT, BeOS, HP MPE, NextStep, PalmPilot, Rhapsody, Windows 2000, Dedicated, OS/390, VSE, TPF, Windows (R) Me, Caldera Open UNIX, OpenBSD, Not Applicable, Windows XP, z/OS, Windows Vista.

Attributes	*Type*
MaximumVersion	String
MinimumVersion	String
TargetOperatingSystem {enum}	uint16

7.1.8.33 CIM_OperatingSystem Class

An OperatingSystem is software/firmware that makes a ComputerSystem's hardware usable, and implements and/or manages the resources, file systems, processes, user interfaces, services available on the ComputerSystem.

Issue 11519 - Vista

Its attribute OSType is an integer indicating the type of OperatingSystem in the range: { "Unknown," "Other," "MACOS," "ATTUNIX," "DGUX," "DECNT," "Tru64 UNIX," "OpenVMS," "HPUX," "AIX," "MVS," "OS400," "OS/2," "JavaVM," "MSDOS," "WIN3x," "WIN95," "WIN98," "WINNT," "WINCE," "NCR3000," "NetWare," "OSF," "DC/OS," "Reliant UNIX," "SCO UnixWare," "SCO OpenServer," "Sequent," "IRIX," "Solaris," "SunOS," "U6000," "ASERIES," "HP NonStop OS," "HP NonStop OSS," "BS2000," "LINUX," "Lynx," "XENIX," "VM," "Interactive UNIX," "BSDUNIX," "FreeBSD," "NetBSD," "GNU Hurd," "OS9," "MACH Kernel," "Inferno," "QNX," "EPOC," "IxWorks," "VxWorks," "MiNT," "BeOS," "HP MPE," "NextStep," "PalmPilot," "Rhapsody," "Windows 2000," "Dedicated," "OS/390," "VSE," "TPF," "Windows (R) Me," "Caldera Open UNIX," "OpenBSD," "Not Applicable," "Windows XP," "z/OS," "Microsoft Windows Server 2003," "Microsoft Windows Server 2003 64-Bit.", "Windows Vista" }

Attributes	*Type*
OSType	String

7.1.8.34 CIM_PhysicalElement Class

This class belongs to the "Hardware System Management" profile.

Subclasses of CIM_PhysicalElement define any component of a System that has a distinct physical identity. Instances of this class can be defined in terms of labels that can be physically attached to the object. All Processes, Files, and LogicalDevices are considered not to be Physical Elements. For example, it is not possible to attach a label to a modem. It is only possible to attach a label to the card that implements the modem. The same card could also implement a LAN adapter. These are tangible Managed SystemElements (usually actual hardware items) that have a physical manifestation of some sort. A ManagedSystem Element is not necessarily a discrete component. For example, it is possible for a single Card (which is a type of PhysicalElement) to host more than one LogicalDevice. The card would be represented by a single PhysicalElement associated with multiple Devices.

Associations	*Multiplicity*	*Class*
CIM_Realizes	* .. *	CIM_LogicalDevice

CIM_PhysicalElementLocation	0 .. 1	CIM_Location

7.1.8.35 CIM_PowerSupply Class

This class belongs to the "Hardware System Management" profile.

Capabilities and management of the PowerSupply LogicalDevice.

7.1.8.36 CIM_Process Class

Each instance of the CIM_Process class represents a single instance of a running program. A user of the OperatingSystem will typically see a Process as an application or task. Within an OperatingSystem, a Process is defined by a workspace of memory resources and environmental settings that are allocated to it. On a multitasking System, this workspace prevents intrusion of resources by other Processes. Additionally, a Process can execute as multiple Threads, all which run within the same workspace. The attributes are:

Issue 11507 - Clarification of Handle on Process and Thread

- "Handle:" A string used to identify the Process. On POSIX systems, this attribute is the process ID. On Win32 systems, this attribute is the process handle.
- "OSName:" The scoping OperatingSystem's Name.

Attributes	*Type*	
OSName	String	
Handle {key}	String	
Associations	*Multiplicity*	*Class*
CIM_ServiceProcess	0 .. *	CIM_Service
CIM_ProcessThread	0 .. *	CIM_Thread

7.1.8.37 CIM_ProcessIndication Class

An abstract superclass for specialized Indication classes, addressing specific changes and alerts published by providers and instrumentation. Subclasses include AlertIndication (with properties such as PerceivedSeverity and ProbableCause), and SNMPTrapIndication (which recasts Traps as CIM indications).

7.1.8.38 CIM_Processor Class

This class belongs to the "Hardware System Management" profile.

Capabilities and management of the Processor LogicalDevice.

The attributes are:

- "LoadPercentage:" Loading of this Processor, averaged over the last minute, in Percent.
- "CPUStatus:" The CPUStatus property indicates the current status of the Processor. Values: 0 (Unknown), 1 (CPU

Enabled), 2 (CPU Disabled by User via BIOS Setup), 3 (CPU Disabled By BIOS (POST Error)), 4 (CPU Is Idle), 7 (Other).

7.1.8.39 CIM_ProtocolEndPoint Class

This class belongs to the "Hardware System Management" profile.

A communication point from which data may be sent or received. CIM_ProtocolEndpoints link system/computer interfaces to LogicalNetworks.

The CIM_EndpointIdentity association indicates that two CIM_ProtocolEndpoints represent different aspects of the same underlying address or protocol-specific ID. This association refines the CIM_LogicalIdentity superclass by restricting it to the Endpoint level and defining its use in well understood scenarios. One of these scenarios is to represent that an Endpoint has both 'LAN' and protocol-specific aspects. For example, an Endpoint could be both a LANEndpoint as well as a DHCPEndpoint.

Associations	*Multiplicity*	*Class*
CIM_MemberOfCollection {key}	1 .. 1	CIM_ConnectivityCollection
CIM_EndpointIdentity	* .. *	CIM_ProtocolEndPoint
CIM_EndpointIdentity	* .. *	CIM_ProtocolEndPoint
CIM_RouteUsesEndpoint	* .. *	CIM_NextHopRoute

7.1.8.40 CIM_RecordForLog Class

The CIM_RecordForLog class is used to instantiate records to be aggregated to a Log.

The attributes are:

- "RecordFormat:" A string describing the data structure of the information in the property, RecordData. If the RecordFormat string is <empty>, RecordData should be interpreted as a free-form string. To describe the data structure of RecordData, the RecordFormat string should be constructed as follows: - The first character is a delimiter character and is used to parse the remainder of the string into sub-strings. - Each sub-string is separated by the delimiter character and should be in the form of a CIM property declaration (i.e., datatype and property name). This set of declarations may be used to interpret the similarly delimited RecordData property. For example, using a '*' delimiter, RecordFormat %3D "*string ThisDay*uint32 ThisYear*datetime SomeTime" may be used to interpret: RecordData %3D "*This is Friday*2002*20020807141000.000000-300."
- "RecordData:" A string containing LogRecord data. If the corresponding RecordFormat property is <empty>, or cannot be parsed according to the recommended format, RecordData should be interpreted as a free-form string. If the RecordFormat property contains parseable format information (as recommended in the RecordFormat Description qualifier), the RecordData string SHOULD be parsed in accordance with this format. In this case, RecordData SHOULD begin with the delimiter character and this character SHOULD be used to separate substrings in the manner described. The RecordData string can then be parsed by the data consumer and appropriately typed.
- "Locale:" A locale indicates a particular geographical, political, or cultural region. The Locale specifies the language used in creating the RecordForLog data. If the Locale property is empty, it is assumed that the default locale is en_US (English). The locale string consists of three sub-strings, separated by underscores:
    - The first sub-string is the language code, as specified in ISO639.

- The second sub-string is the country code, as specified in ISO3166.
- The third sub-string is a variant, which is vendor specific.

For example, US English appears as: "en_US_WIN," where the "WIN" variant would specify a Windows browser-specific collation (if one exists). Since the variant is not standardized, it is not commonly used and generally is limited to easily recognizable values ("WIN," "UNIX," "EURO," etc.) used in standard environments. The language and country codes are required; the variant may be empty.

Attributes	*Type*
RecordFormat	String
RecordData	String
Locale	String

7.1.8.41 CIM_RedundancyGroup Class

This class belongs to the "Fault Tolerance Management" and "Load Balancing Management" profiles.

A class derived from CIM_LogicalElement that is a special collection of CIM_ManagedSystemElements. This collection indicates that the aggregated components together provide redundancy. All elements aggregated in a CIM_RedundancyGroup should be instantiations of the same object class.

7.1.8.42 CIM_RemoteServiceAccessPoint Class

This class belongs to the "Hardware System Management" profile.

CIM_RemoteServiceAccessPoint describes access and/or addressing information for a remote connection, that is known to a 'local' network element. This information is scoped/contained by the 'local' network element, since this is the context in which it is 'remote.' The attributes are:

- "AccessInfo:" Access and/or addressing information for a remote connection. This can be a host name, network address, or similar information.
- "InfoFormat:" An enumerated integer describing the format and interpretation of the AccessInfo property. The enumeration is: Other, Host Name, IPv4 Address, IPv6 Address, IPX Address, DECnet Address, SNA Address, Autonomous System Number, MPLS Label, IPv4 Subnet Address, IPv6 Subnet Address, IPv4 Address Range, IPv6 Address Range, Dial String, Ethernet Address, Token Ring Address, ATM Address, Frame Relay Address, URL, FQDN, User FQDN, DER ASN1 DN, DER ASN1 GN, Key ID, DMTF Reserved, Vendor Reserved.

Attributes	*Type*	
AccessInfo	String	
InfoFormat	uint16	
Associations	*Multiplicity*	*Class*
CIM_AssociatedNextHop	* .. *	CIM_NextHopRoute

7.1.8.43 CIM_Sensor Class

This class belongs to the "Hardware System Management" profile. A CIM_Sensor is a hardware device capable of measuring the characteristics of some physical property (for example, the temperature or voltage characteristics of a CIM_Computer System).

7.1.8.44 CIM_Service Class

A CIM_Service is a Logical Element that contains the information that is necessary to represent and manage the functionality provided by a Device or a SoftwareFeature, or both. A Service is a general-purpose object that is used to configure and manage the implementation of functionality. It is not the functionality itself.

7.1.8.45 CIM_ServiceAccessPoint Class

This class belongs to the "Hardware System Management" profile. CIM_ServiceAccessPoint represents the ability to utilize or invoke a Service. Access points represent that a Service is made available to other entities for use. The attributes are:

- "Name:" The Name property uniquely identifies the ServiceAccessPoint and provides an indication of the functionality that is managed.

Attributes	*Type*	
Name {key}	String	
Associations	*Multiplicity*	*Class*
CIM_HostedAccessPoint {key} {override}	1 .. 1	AMS_ComputerSystem
CIM_BindsToLANEndPoint	* .. *	CIM_LANEndPoint

7.1.8.46 CIM_SoftwareElement Class

The CIM_SoftwareElement class is used to decompose a CIM_SoftwareFeature object into a set of individually manageable or deployable parts, for a particular platform. A SoftwareElement’s platform is uniquely identified by its underlying hardware architecture and operating system (for example Sun Solaris on Sun Sparc or Windows NT on Intel platforms). As such, to understand the details of how the functionality of a particular SoftwareFeature is provided on a particular platform, the CIM_SoftwareElement objects referenced by CIM_SoftwareFeatureSoftwareElements associations are organized in disjoint sets.

7.1.8.47 CIM_SoftwareFeature Class

The CIM_SoftwareFeature class defines a particular function or capability of a product or application system. This class captures a level of granularity describing a unit of installation, rather than the units that reflect how the product is built or packaged. The latter detail is captured using a CIM_SoftwareElement class. When a SoftwareFeature can exist on multiple platforms or operating systems (for example, a client component of a three tiered client/server application that runs on Solaris, Windows NT, and Windows 95), the Feature is a collection of all the SoftwareElements for these different platforms. In this case, the users of the model must be aware of this situation since typically they will be interested in a sub-collection of the SoftwareElements required for a particular platform.

7.1.8.48 CIM_StorageExtent Class

This class belongs to the "Hardware System Management" profile.

StorageExtent describes the capabilities and management of the various media that exist to store data and allow data retrieval. This superclass could be used to represent the various components of RAID (Hardware or Software) or as a raw logical extent on top of physical media.

7.1.8.49 CIM_System Class

CIM_System represents an entity made up of component parts (defined by the CIM_SystemComponent relationship), that operates as a 'functional whole.' Systems are top level objects in the CIM hierarchy, requiring no scoping or weak relationships in order to exist and have context. It should be reasonable to uniquely name and manage a System at an enterprise level. For example, a CIM_ComputerSystem is a kind of CIM_System that can be uniquely named and independently managed in an enterprise. However, this is not true for the power supply (or the power supply sub-'system') within the computer.

Although a System may be viewed as a Collection, this is not the correct model. A Collection is simply a 'bag' that 'holds' its members. A System is a higher level abstraction, built out of its individual components. It is more than a sum of its parts. Note that System is a subclass of EnabledLogicalElement which allows the entire abstraction to be functionally enabled/disabled - at a higher level than enabling/disabling its component parts.

Associations	*Multiplicity*	*Class*
CIM_ServiceProcess	0 .. *	CIM_Service

7.1.8.50 CIM_Thread Class

Issue 11507 - Clarification of Handle on Process and Thread

Threads represent the ability to execute units of a Process or task in parallel. A Process can have many Threads, each of which is weak to the Process.

The attributes are:

- "Handle:" A string used to identify the Thread. On POSIX systems, this attribute is the thread ID. On Win32 systems, this attribute is the thread handle.
- "OSName:" The scoping OperatingSystem's Name.
- "Priority:" Priority indicates the urgency or importance of execution of a Thread. A Thread may have a different priority than its owning Process. If this information is not available for a Thread, a value of 0 should be used.

Attributes	*Type*	
OSName	String	
Handle {key}	String	
Priority	uint32	
Associations	*Multiplicity*	*Class*

CIM_ProcessThread	1	CIM_Process

7.1.8.51 CIM_UnixProcess Class

This class is specific to POSIX systems.

Each instance of the CIM_UnixProcess class represents a single instance of a running program. A user of the Operating System will typically see a Process as an application or task. Within an OperatingSystem, a Process is defined by a workspace of memory resources and environmental settings that are allocated to it. On a multitasking System, this workspace prevents intrusion of resources by other Processes. Additionally, a Process can execute as multiple Threads, all which run within the same workspace.

The attributes are:

- "ProcessGroupID:" The Group ID of this currently executing process.
- "RealUserID:" The Real User ID of this currently executing process.
- "Parameters:" The operating system parameters provided to the executing process. These are the argv[] values.

Attributes	*Type*
ProcessGroupID	uint64
RealUserID	uint64
Parameters	list<string>

7.1.8.52 CIM_UnixThread Class

This class is specific to POSIX systems.

Threads represent the ability to execute units of a Process or task in parallel. A UnixThread inherits from the superclass, CIM_Thread, which is weak to the Process.

7.1.8.53 CIM_Watchdog Class

This class belongs to the "Hardware System Management" profile.

CIM_Watchdog is a timer implemented in system hardware. It allows the hardware to monitor the state of the Operating System, BIOS or a software component installed on the System. If the monitored component fails to re-arm the timer before its expiration, the hardware assumes that the System is in a critical state, and could reset the ComputerSystem.

7.1.9 Lightweight Logging Service Package

This package gathers Lightweight Logging Service classes used by the AMSM service. The classes are a proper subset of classes defined by the OMG standard. The documentation shown afterward is an extract from [LWLOG].

7.1.9.1 AdministrativeState Class

The AdministrativeState class denotes the active logging state of an operational Log. When set to UNLOCKED the Log will accept records for storage, per its operational parameters. When set to LOCKED the Log will not accept new log records and records can be read or deleted only.

Enumeration
LOCKED
UNLOCKED

7.1.9.2 AvailibilityStatus Class

The AvailabilityStatus denotes whether or not the Log is available for use. When true, offDuty indicates the Log is LOCKED (administrative state) or DISABLED (operational state). When true, logFull indicates the Log storage is full.

Attributes	*Type*
offDuty	boolean
logFull	boolean

7.1.9.3 LogAdministrator Class

Operations	*Parameters*	*Parameters type*
setMaxSize	return type	uint16
	in size	uint32
setLogFullAction	return type	uint16
	in action	LogFullAction
setAdministrativeState	return type	uint16
	in state	AdministrativeState
clearLog	return type	uint16
Destroy	return type	uint16

The following sections review the operation in detail.

7.1.9.3.1 Destroy ( )

Tear down an instantiated Log.

7.1.9.3.2 clearLog ( )

Purge the log storage area.

7.1.9.3.3 setAdministrativeState (in state: AdministrativeState)

The setAdministrativeState operation provides write access to the administrative state value.

7.1.9.3.4 setLogFullAction (in action: LogFullAction)

Configure the action to be taken if the log storage area becomes full.

7.1.9.3.5 setMaxSize (in size: uint32)

Sets the maximum size the Log storage area.

7.1.9.4 LogConsumer Class

Operations	*Parameters*	*Parameters type*
getRecordIdFromTime	return type	uint64
	in fromTime	datetime
retrieveById	return type	LogRecordSequence
	inout currentId	uint64
	inout howMany	uint16

The following sections review the operation in detail.

7.1.9.4.1 getRecordIdFromTime (in fromTime: datetime)

Identify a record in the log a record based on its time stamp.

The getRecordIdFromTime operation returns the record Id of the first record in the Log with a time stamp that is greater than, or equal to, the time specified in the fromTime parameter. If the Log does not contain a record that meets the criteria provided, then the RecordId returned corresponds to the next record that will be recorded in the future. In this way, if this "future" recordId is passed into the retrieveById operation, an empty record will be returned unless records have been recorded since the time specified. Note that if the time specified in the fromTime parameter is in the future, there is no guarantee that the resulting records returned by the retrieveById operation will have a time stamp after the fromTime parameter if the returned recordId from this invocation of the getRecordIdFromTime operation is subsequently used as input to the retrieveById operation.

7.1.9.4.2 retrieveById (inout currentId: uint64, inout howMany: uint16)

Retrieves a specified number of records from the Log.

The retrieveById operation returns a LogRecordSequence that begins with the record specified by the currentId parameter. The number of records in the LogRecordSequence returned by the retrieveById operation is equal to the number of records specified by the howMany parameter, or the number of records available if the number of records

specified by the howMany parameter cannot be met. The log will update howMany to indicate the number of records returned and will set currentId to the id of the record following the last retrieved record. If there are no further records available, currentId will be set to zero. If the record specified by currentId does not exist, or if the Log is empty, the retrieveById operation returns an empty list of LogRecords, and sets both, currentId, and howMany to zero.

7.1.9.5 LogFullAction Class

This type specifies the action that the Log should take when its internal buffers become full of data, leaving no room for new records to be written. WRAP indicates that the Log will overwrite the oldest LogRecords with the newest records, as they are written to the Log. HALT indicates that the Log will stop logging when full.

Enumeration
WRAP
HALT

7.1.9.6 LogRecord Class

The LogRecord Type defines the format of the LogRecords as stored in the Log. It represents an encapsulation of the ProducerLogRecord, supplied by the log producer, and adds the time stamp (via the LogTime structure) and a unique record identification (via the RecordID field).

Attributes	*Type*	
id	uint64	
time	datetime	
Associations	*Multiplicity*	*Class*
CIM_LogicalIdentity	1 .. 1	AMS_LogRecord
	1 .. 1	ProducerLogRecord

7.1.9.7 LogRecordSequence Class

The LogRecordSequence type defines an unbounded sequence of LogRecords.

Associations	*Multiplicity*	*Class*
	* .. *	LogRecord

7.1.9.8 LogStatus Class

Interface LogStatus provides access to operations of common interest, which are available through inheritance in all interfaces of the logging service.

Operations	*Parameters*	*Parameters type*
getMaxSize	return type	uint32

getCurrentSize	return type	uint32
getNumRecords	return type	uint32
getLogFullAction	return type	LogFullAction
getAvailibilityStatus	return type	AvailibilityStatus
getAdministrativeState	return type	AdministrativeState
getOperationalState	return type	OperationalState

The following sections review the operation in detail.

7.1.9.8.1 getAdministrativeState ( )

Returns the administrative state of the Log.

The ability of the logging service to accept and store new logging records can be affected by administrative action. The getAdministrativeState is used to read the administrative state of the Log. The possible states are LOCKED and UNLOCKED. If the state is LOCKED, no new records are accepted. Reading of already stored records is not affected.

7.1.9.8.2 getAvailibilityStatus ( )

Returns the availability status of the Log.

The ability of the Log to accept and store logging records might become impaired. The getAvailabilityStatus operation is used to check the availability status of the Log. The returned instance of the AvailibilityStatus type contains two Boolean values: offDuty, which indicates the log is disabled when true, and logFull, which indicates that all free space is depleted in the log storage area.

7.1.9.8.3 getCurrentSize ( )

Returns the amount of log storage area currently occupied by logging records.

7.1.9.8.4 getLogFullAction ( )

Returns the action to be taken when the storage area becomes full.

Since the storage space of the Log storage area is finite, the Logging Service has to take special action when the free space is depleted. The kind of action is described by the LogFullAction type. The getLogFullAction operation returns the information about which action the Logging Service will take when the storage area becomes full. The possible values are HALT, which means no further logging records are accepted and stored; or WRAP, which means the log continues by overwriting the oldest records in the storage area.

7.1.9.8.5 getMaxSize ( )

Returns the size of the logging storage area.

7.1.9.8.6 getNumRecords ( )

Returns the number of records presently stored in the Log.

7.1.9.8.7 getOperationalState ( )

Returns the operational state of the Log.

The getOperationalState operation returns the actual operational state of the log. Possible values are ENABLED, which means the log is fully functional and available to log producer and log consumer clients; or DISABLED, which indicates the log has encountered a runtime problem and is not available for use by log producers or log consumers.

7.1.9.9 OperationalState Class

The enumeration OperationalState defines the Log states of operation. When the Log is ENABLED it is fully functional and available for use by log producer and log consumer clients. A Log that is DISABLED has encountered a runtime problem and is not available for use by log producers or log consumers. The internal error conditions that cause the Log to go into DISABLED state are implementation specific.

7.1.9.10 ProducerLogRecord Class

The ProducerLogRecord represents the log record written by the log producer client to the log. It will be encapsulated by in a LogRecord object before it is stored in the log storage area.

Attributes	*Type*
producerId	String
producerName	String
level	uint16
logData	String

7.1.10 Logical Hardware Package

It represents the "Hardware" sub-package, which groups the classes describing the effective hardware topology. These classes permit the representation of an actual network. The essential class is AMS_ComputerSystem which represents:

- A computer as an aggregation of hardware elements.
- A computer as a node in a network.

Hence, an AMS_ComputerSystem aggregates a hardware configuration (CIM_LogicalDevice) - processor, memory, file systems, and gets some operating systems (AMS_OperatingSystem), which supports application models (cf. "Supported Application Model Package").

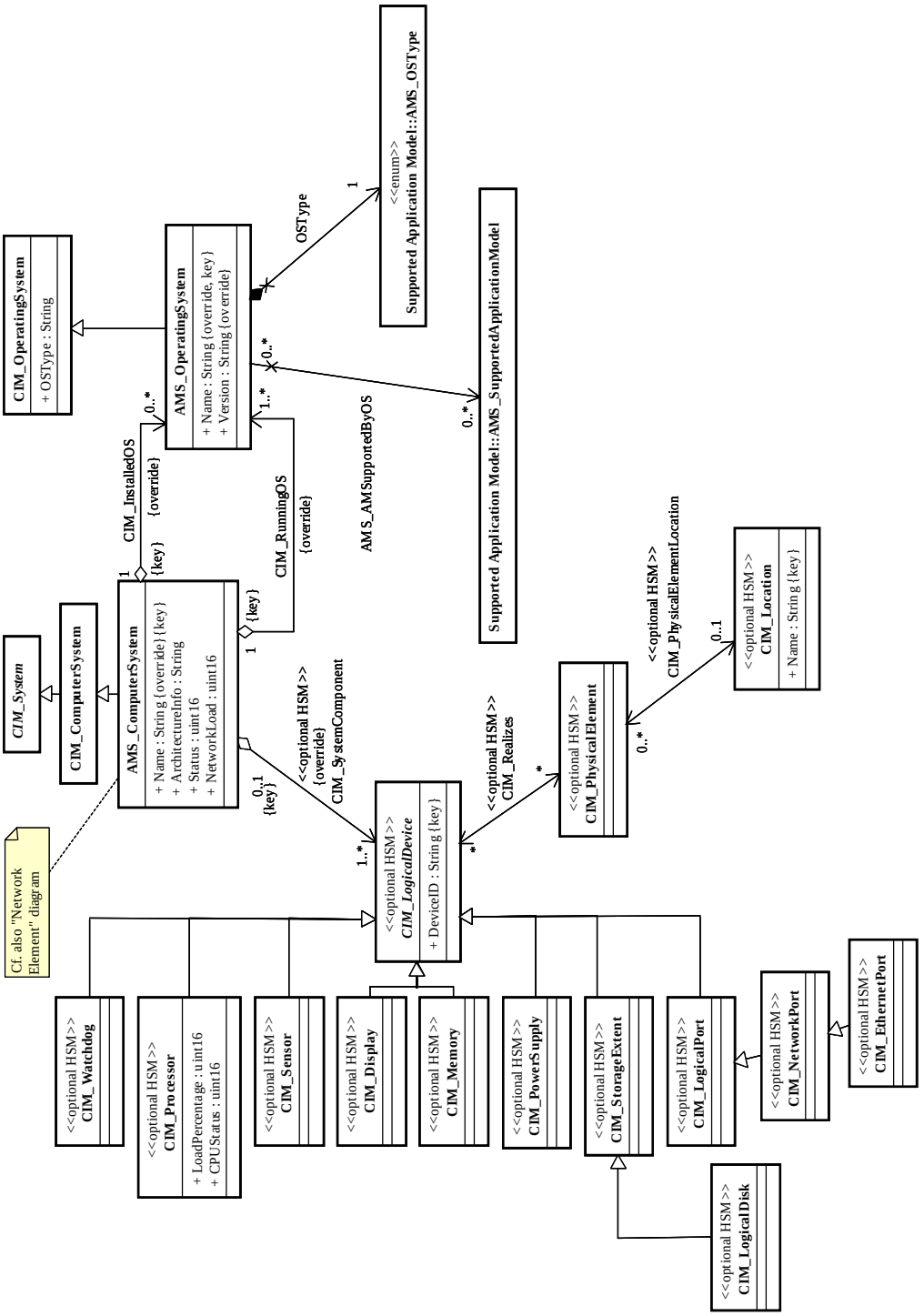


Figure 7.19 - Computer System class diagram

Moreover, it has access points (AMS_ProtocolEndPoint), i.e., Ethernet ports and route tables.

Eventually, AMS_ComputerSystems are logically organized through domains (AMS_Domain) or hardware groups (AMS_HardwareGroup).

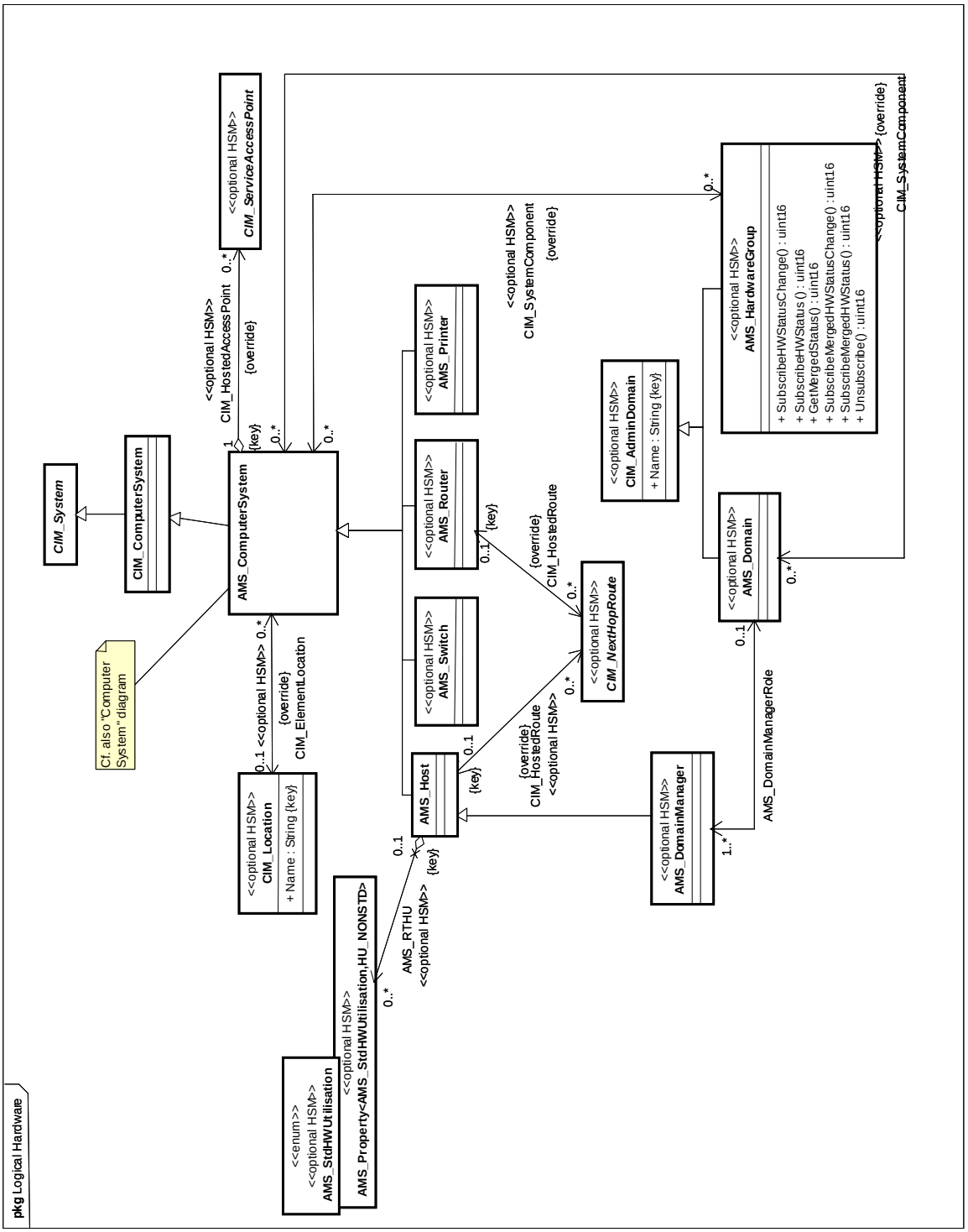


Figure 7.20 - Network class diagram (1/2)

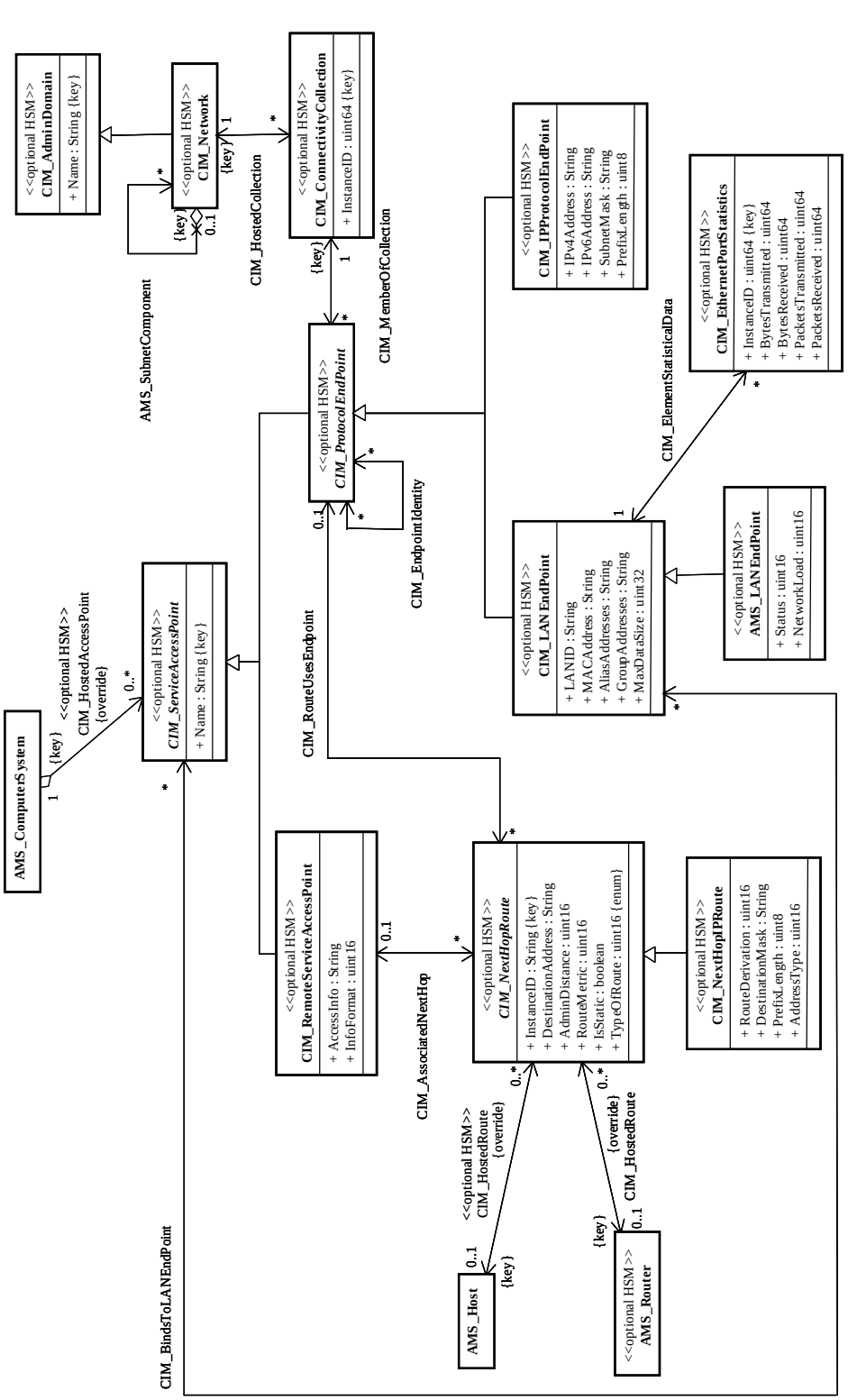


Figure 7.21 - Network class diagram (2/2)

All the AMS_ComputerSystem class and sub-classes are interfaces offering monitoring methods. The following instantiation diagram shows an example of such a network.

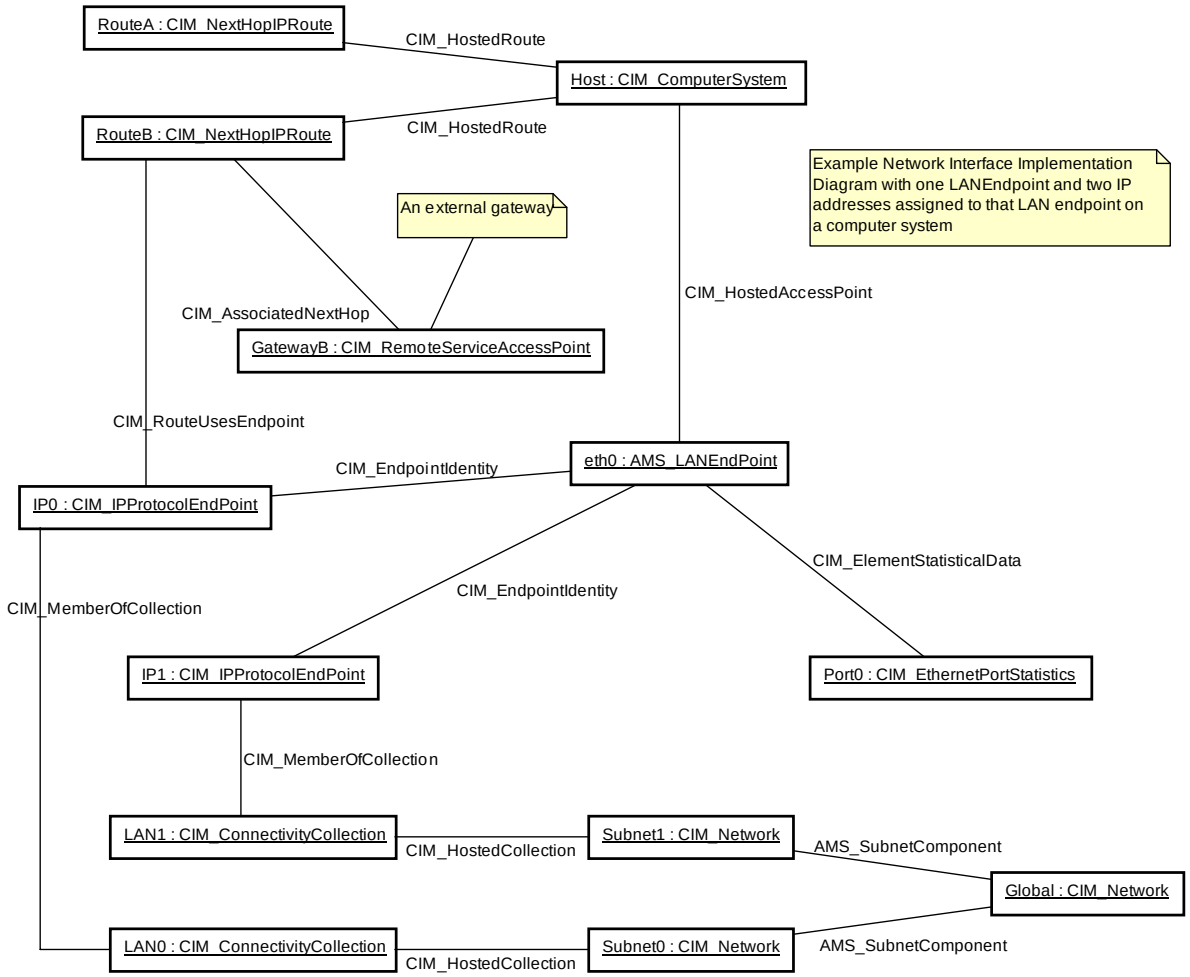


Figure 7.22 - Network instantiation

7.1.10.1 AMS_ComputerSystem Class

The AMS_ComputerSystem class models the hosts. Its subclasses specify which kind of elements are modeled: hosts (AMS_Host), routers (AMS_Router), switches (AMS_Switch), or printers (AMS_Printer).

It aggregates some CIM_LogicalDevice which is "an abstraction or emulation of a hardware entity: CIM_Processor, CIM_Memory, etc." (CIM documentation). The actual type of element is described in one of the subclasses of CIM_LogicalDevice allowed by CIM.

It aggregates also some AMS_OperatingSystems as CIM_RunningOS and CIM_InstalledOS. On account of so-called virtualisation, several operating system may be running on an unique AMS_ComputerSystem, hence the multiplicity of the CIM_RunningOS association is 1..*.

It may be associated with a physical location (CIM_Location), with some service access points (CIM_ServiceAccessPoint).

The CIM_ServiceAccessPoint class does not preclude any type of protocol. The actual protocol may be specified with one of its subclasses. For instance the CIM_IPProtocolEndPoint defines end points running on IP with IPv4 or IPv6 addresses and a subnet mask.

The attributes of the AMS_ComputerSystem are:

- A name that is the name of the host.
- ArchitectureInfo that specify some textual information on the architecture of the element.
- Status that is the state of the computer. The possible values are implementation-dependant.
- NetworkLoad that is the global network load on this host.

Attributes	*Type*	
Name {override} {key}	String	
ArchitectureInfo	String	
Status	uint16	
NetworkLoad	uint16	
Associations	*Multiplicity*	*Class*
CIM_HostedAccessPoint {override} <<optional HSM>>	0 .. *	CIM_ServiceAccessPoint
CIM_SystemComponent {override} <<optional HSM>>	1 .. *	CIM_LogicalDevice
CIM_ElementLocation {override} <<optional HSM>>	0 .. 1	CIM_Location
CIM_SystemComponent {override} <<optional HSM>>	0 .. *	AMS_Domain
CIM_InstalledOS {override}	0 .. *	AMS_OperatingSystem
CIM_RunningOS {override}	1 .. *	AMS_OperatingSystem
CIM_SystemComponent {override} <<optional HSM>>	0 .. *	AMS_HardwareGroup
AMS_ConfSpecCS <<optional HSM>>	0 .. 1	AMS_ConfigurationSpecification

7.1.10.2 AMS_Domain Class

This class belongs to the "Hardware System Management" profile. The AMS_Domain class models a set of computers in a single management domain. It has a set of AMS_ComputerSystem, some AMS_DomainManagers (at least one) and an attribute giving its name.

Associations	*Multiplicity*	*Class*
AMS_DomainManagerRole	1 .. *	AMS_DomainManager
CIM_SystemComponent {override}	0 .. *	AMS_ComputerSystem
AMS_ConfSpecDom	0 .. 1	AMS_ConfigurationSpecification

7.1.10.3 AMS_DomainManager Class

This class belongs to the "Hardware System Management" profile.

The AMS_DomainManager class represents a computer node in a system which is responsible for managing a AMS_Domain. There may be one or more AMS_DomainManager instances for a given AMS_Domain.

AMS_DomainManager is derived from the AMS_Host class.

It may be associated with an AMS_Domain.

Associations	*Multiplicity*	*Class*
AMS_DomainManagerRole	0 .. 1	AMS_Domain

7.1.10.4 AMS_HardwareGroup Class

This class belongs to the "Hardware System Management" profile. The AMS_HardwareGroup class represents groupings of AMS_ComputerSystems. It has a set of AMS_ComputerSystems and an attribute giving its name.

Associations	*Multiplicity*	*Class*
CIM_SystemComponent {override}	0 .. *	AMS_ComputerSystem
AMS_ConfSpecHG	0 .. 1	AMS_ConfigurationSpecification
Operations	*Parameters*	*Parameters type*
SubscribeHWStatusChange	return type	uint16
	out subscriptionID	uint32
SubscribeHWStatus	return type	uint16
	delay	uint16
	out subscriptionID	uint32
GetMergedStatus	return type	uint16

SubscribeMergedHWStatusChange	return type	uint16
	out subscriptionID	uint32
SubscribeMergedHWStatus	return type	uint16
	delay	uint16
	out subscriptionID	uint32
Unsubscribe	return type	uint16
	subscriptionID	uint32

The following sections review the operation in detail.

7.1.10.4.1 GetMergedStatus ( )

This operation gets the merged status computed from the status of all the Computer Systems of the Hardware Group.

The actual algorithm is implementation dependant.

7.1.10.4.2 SubscribeHWStatus (delay: uint16, out subscriptionID : uint32)

This operation subscribes to receive periodically the status of the hardware items gathered in the Hardware Group.

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The data returned are a collection of AMS_RTHWIndication.

7.1.10.4.3 SubscribeHWStatusChange (out subscriptionID : uint32)

This operation subscribes to the modifications of the status of the hardware items gathered in the Hardware Group.

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The data returned are a collection of AMS_RTHWIndication.

7.1.10.4.4 SubscribeMergedHWStatus (delay: uint16, out subscriptionID : uint32)

This operation subscribes to receive periodically the merged status of the Hardware Group (cf. GetMergedStatus).

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The data returned are a collection of AMS_RTHWIndication.

7.1.10.4.5 SubscribeMergedHWStatusChange (out subscriptionID : uint32)

This operation subscribes to the modifications of the merged status of the Hardware Group (cf. GetMergedStatus).

“subscriptionID” is the ID to be passed to the corresponding call to unsubscribe.

The data returned are a collection of AMS_RTHWIndication.

7.1.10.4.6 Unsubscribe (subscriptionID: uint32)

This operation deletes a previous subscription demand.

This operation shall return AMS_BADSUBSCRIPTIONID if the parameter is erroneous..

7.1.10.5 AMS_Host Class

The AMS_Host class models the hosts as points in a network.

It is a subclass of AMS_ComputerSystem and a superclass of AMS_DomainManager.

Since a host can be used for routing, it aggregates some routes (CIM_NextHopRoute). The CIM_NextHopRoute class does not preclude any type of protocol. The actual protocol may be specified with one of its subclasses.

It has also an association to AMS_Property<AMS_StdHWUtilisation,HU_NONSTD> in order to design platform-specific hardware utilizations (cf. AMS_StdHWUtilisation for the normalized set of platform-specific hardware utilizations).

Associations	*Multiplicity*	*Class*
CIM_HostedRoute {override} <<optional HSM>>	0 .. *	CIM_NextHopRoute
AMS_RTHU <<optional HSM>>	1 .. 1	AMS_Property<AMS_StdHWUtilisation,HU_NONSTD>
AMS_HostUsed	0 .. *	AMS_DeploymentLink

7.1.10.6 AMS_LANEndPoint Class

This class belongs to the "Hardware System Management" profile.

The AMS_LANEndPoint class subclasses CIM_LANEndPoint in order to add the following attributes:

- The status of the communication end point (up or down).
- The network load on this end point (transmission rate expressed in bytes per seconds).

Attributes	*Type*	
Status	uint16	
NetworkLoad	uint16	

7.1.10.7 AMS_OperatingSystem Class

The class AMS_OperatingSystem models the operating system on hosts. Such an operating system belongs to an AMS_ComputerSystem either as the running operating system, or as one of the installed operating systems. It is associated with some supported application models (cf. AMS_SupportedApplicationModel). Its attributes are:

- a name that, with its AMS_ComputerSystem's global name, takes part of the make-up of its global name.

- a version string that must include any patch information.

Attributes	*Type*	
Name {override, key}	String	
Version {override}	String	
Associations	*Multiplicity*	*Class*
CIM_InstalledOS {key} {override}	1 .. 1	AMS_ComputerSystem
CIM_RunningOS {key} {override}	1 .. 1	AMS_ComputerSystem
AMS_AMSupportedByOS	0 .. *	AMS_SupportedApplicationModel
OSType	1 .. 1	AMS_OSType
AMS_ConfSpecOS <<optional HSM>>	0 .. 1	AMS_ConfigurationSpecification

7.1.10.8 AMS_Printer Class

This class belongs to the "Hardware System Management" profile. The AMS_Printer class models printers as points in a network. It is a subclass of AMS_ComputerSystem.

7.1.10.9 AMS_Router Class

This class belongs to the "Hardware System Management" profile. The AMS_Router class models routers as points in a network. It is a subclass of AMS_ComputerSystem. It aggregates some routes (CIM_NextHopRoute). The CIM_NextHopRoute class does not preclude any type of protocol. The actual protocol may be specified with one of its subclasses.

Associations	*Multiplicity*	*Class*
CIM_HostedRoute {override}	0 .. *	CIM_NextHopRoute

7.1.10.10 AMS_StdHWUtilisation class

The AMS_StdHWUtilisation class enumerates the standardized platform-specific hardware utilizations:

- HU_NONSTD: special value that denotes a non-normalized value (i.e. use the "Name" attribute instead)
- HU_CPU_LOAD: the percentage of time that the CPUs were not idle (uint16)
- HU_NETWORK_LOAD: ratio between occupied Bandwidth and available Bandwidth (float)
- HU_NETWORK_BANDWIDTH: available bandwidth in bits per seconds (uint16)
- HU_VIRTUAL_MEMORY: size in bytes of the virtual memory (uint32)
- HU_VIRTUAL_MEMORY_OCCUPATION: size in bytes of the occupied virtual memory (uint32)
- HU_TOTAL_MEMORY: size in byte of the total memory (Virtual and Physical) (uint32)

- HU_TOTAL_MEMORY_OCCUPATION: size in byte of the total occupied memory (Virtual and Physical) (uint32)
- HU_PROCESS_NUMBER: number of running processes (uint16)
- HU_THREAD_NUMBER: number of running threads (uint16)
- HU_DISK: disk size in bytes (uint64)
- HU_DISK_OCCUPATION: occupied disk size in bytes (uint64)

7.1.10.11 AMS_Switch Class

This class belongs to the "Hardware System Management" profile. The AMS_Switch class models switches as points in a network. It is a subclass of AMS_ComputerSystem.

7.1.11 Logical Hardware Specification Package

This package belongs entirely to the "Hardware System Management" profile. The "Logical Hardware Specification" package groups the classes describing configuration specification for the hardware. A specification of configuration is modeled as a set of name-value pairs.

Each value is defined by either a range of possible values, or a set a possible values, or constraint specified with an implementation-specific constraint language. Each name is defined either in a normalized enumeration, or in an implementation-specific string.

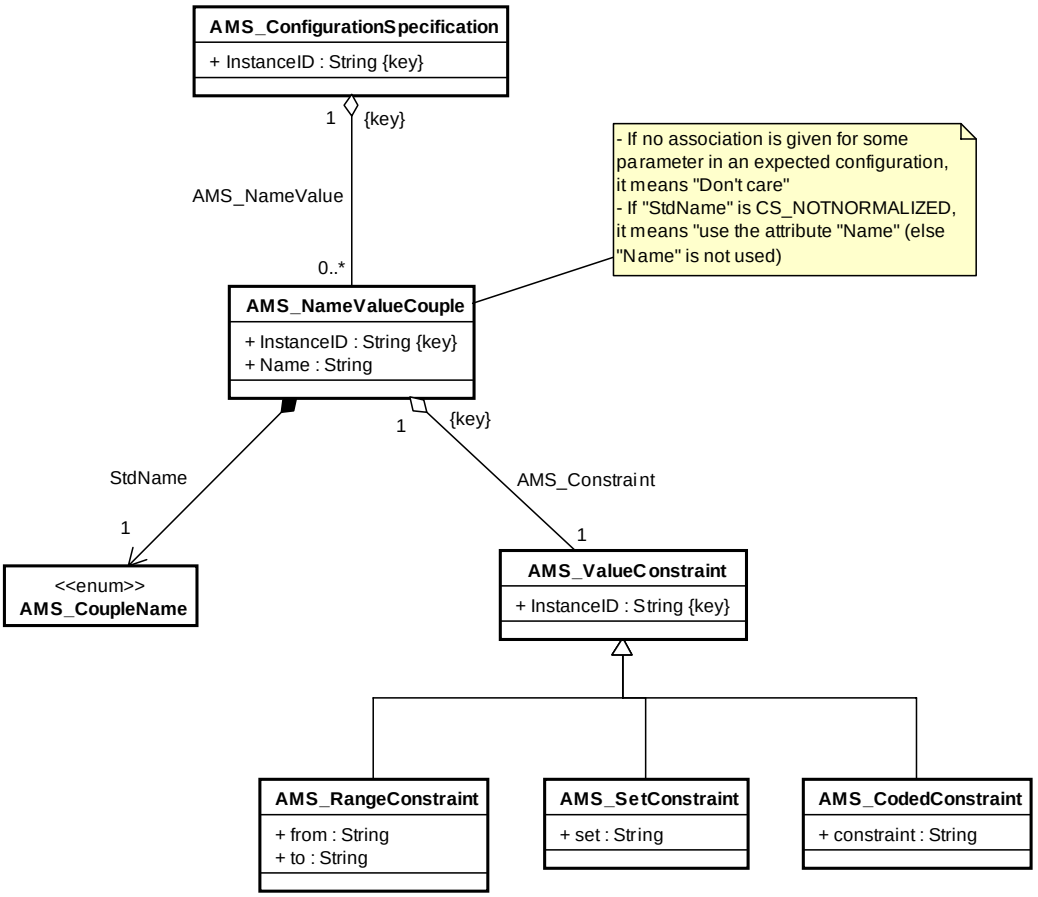


Figure 7.23 - Logical Hardware Specification class diagram (1/2)

AMS_ConfigurationSpecification holds configuration parameters for:

- operating systems (AMS_OperatingSystem)
- computers (AMS_ComputerSystem)
- network domains (AMS_Domain)
- hardware groups (AMS_HardwareGroup)
- deployment specifications (AMS_DeploymentLinkSpec)

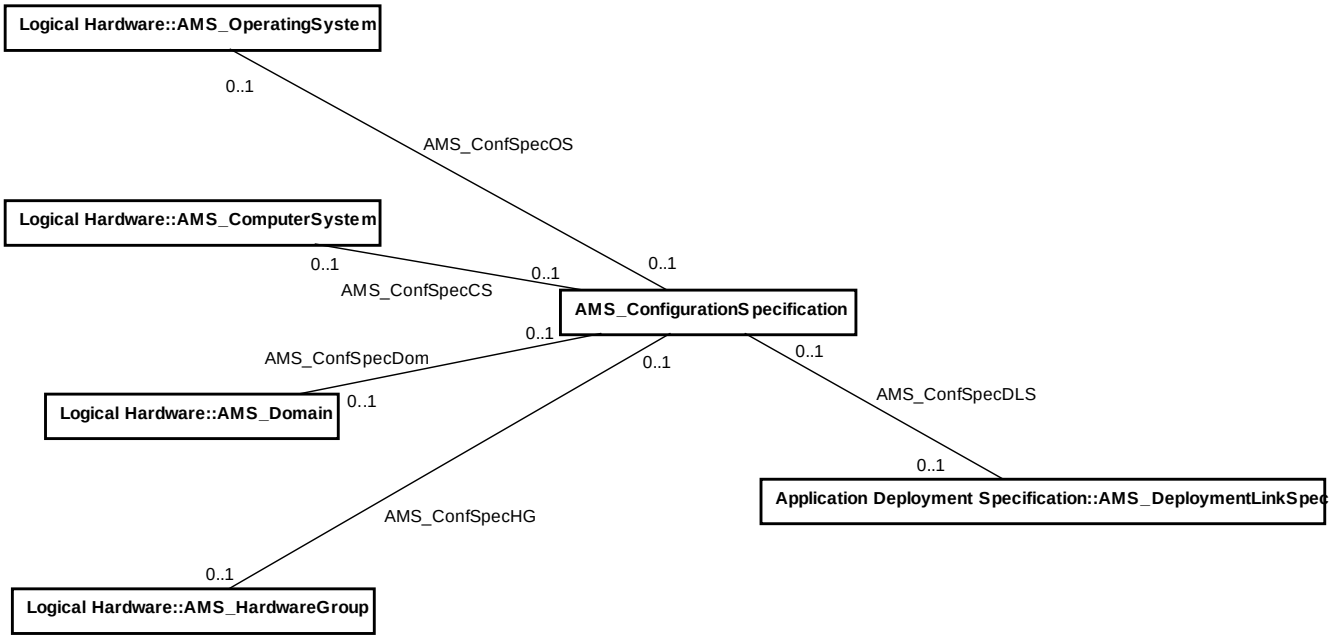


Figure 7.24 - Logical Hardware Specification class diagram (2/2)

7.1.11.1 AMS_CodedConstraint Class

This class belongs to the "Hardware System Management" profile. The AMS_CodedConstraint class models contraints on the values in a specification of configuration by using an implementation-specific language. The constraint is stored in the attribute "constraint".

Attributes	*Type*
constraint	String

7.1.11.2 AMS_ConfigurationSpecification Class

This class belongs to the "Hardware System Management" profile. The AMS_ConfigurationSpecification class models the specifications of configuration. It is a set of parameters modelized by the AMS_NameValueCouple class. If no association is given for some parameter in an expected configuration, it means "Don't care".

Attributes	*Type*	
InstanceID {key}	String	
Associations	*Multiplicity*	*Class*
AMS_ConfSpecCS	0 .. 1	AMS_ComputerSystem
AMS_ConfSpecOS	0 .. 1	AMS_OperatingSystem
AMS_ConfSpecDom	0 .. 1	AMS_Domain

AMS_ConfSpecHG	0 .. 1	AMS_HardwareGroup
AMS_ConfSpecDLS	0 .. 1	AMS_DeploymentLinkSpec
AMS_NameValue	0 .. *	AMS_NameValueCouple

7.1.11.3 AMS_CoupleName Class

This class belongs to the "Hardware System Management" profile.

The AMS_CoupleName enumeration defines the normalized name of parameters in a specification of configuration.

- CS_NOTNORMALIZED points out that the containing "AMS_NameValueCouple" has an implementation-defined name.
- CS_NAME: the associated (string) value is the name of an element - physical element, computer system, operating system, domain (e.g., using IPMI or HPI, name to be used to retrieve detailed information, such as temperature).
- CS_FRU: the associated (Boolean) value indicates if the element correspond to a Field Replaceable Unit ?
- CS_POSITION: the associated (integer) value is the relative position of the physical element w.r.t. the enclosing physical element (e.g., slot number in a chassis).
- CS_INTERFACE: the associated (Boolean) value indicates if this element implement one or more interfaces of the computer system ?
- CS_MFGDATETIME: the associated (string) value is the manufacturing date and time.
- CS_MANUFACTURER: the associated (string) value is the manufacturer name or identification.
- CS_PRODUCTNAME: the associated (string) value is a product name.
- CS_PRODUCTVERSION: the associated (string) value is a product version.
- CS_SERIALNUMBER: the associated (string) value is the serial number.
- CS_PRODUCTTYPE: the associated (string) value is the type of product.
- CS_ASSETTAG: the associated (string) value is any complementary information (e.g., NATO number).
- CS_CHASSISTYPE: the associated (string) value is a type of chassis provided or required (e.g., "ATCA," "CPCI_3U," "CPCI-6U," "VME").
- CS_MACADDRESS: the associated (string) value is a physical hardware (Media Access Control) address.
- CS_POWERSTATE: the associated (string) value is a power state (i.e., Power On or Power Off).
- CS_STATUS: the associated (string) value is a consolidated status (e.g., "Operational," "Failed," "Warning").
- CS_POSTRESULT: the associated (string) value is result of "Power On Self Test" (POST).

Enumeration
CS_NOTNORMALIZED
CS_NAME
CS_FRU
CS_POSITION
CS_INTERFACE
CS_MFGDATETIME
CS_MANUFACTURER
CS_PRODUCTNAME
CS_PRODUCTVERSION
CS_SERIALNUMBER
CS_PRODUCTTYPE
CS_ASSETTAG
CS_CHASSISTYPE
CS_MACADDRESS
CS_POWERSTATE
CS_STATUS
CS_POSTRESULT

7.1.11.4 AMS_NameValueCouple Class

This class belongs to the "Hardware System Management" profile. The AMS_NameValueCouple class models the parameters of a specification collection. It has a constraint on its values (association AMS_Constraint), and a name specified either as a string (attribute Name) or as an enumeration (association "StdName").

If "StdName" has not the special value CS_NOTNORMALIZED, its standardized value must be used.

If "StdName" has the special value CS_NOTNORMALIZED, the attribute "Name" has to be used and its value is implementation-dependant.

Attributes	*Type*	
InstanceID {key}	String	

Name	String	
Associations	*Multiplicity*	*Class*
AMS_NameValue {key}	1 .. 1	AMS_ConfigurationSpecification
StdName	1 .. 1	AMS_CoupleName
AMS_Constraint	1 .. 1	AMS_ValueConstraint

7.1.11.5 AMS_RangeConstraint Class

This class belongs to the "Hardware System Management" profile. The AMS_RangeConstraint class models constraints on the values in a specification of configuration by specifying a lower bound (attribute from) and an upper bound (attribute to).

Attributes	*Type*
from	String
to	String

7.1.11.6 AMS_SetConstraint Class

This class belongs to the "Hardware System Management" profile. The AMS_SetConstraint class models constraints on the values in a specification of configuration by specifying a set of possibilities (attribute "set").

Attributes	*Type*
set	String

7.1.11.7 AMS_ValueConstraint Class

This class belongs to the "Hardware System Management" profile. The AMS_ValueConstraint class models the values in a specification of configuration. It is the upper class for the real constraint class.

Attributes	*Type*	
InstanceID {key}	String	
Associations	*Multiplicity*	*Class*
AMS_Constraint {key}	1 .. 1	AMS_NameValueCouple

7.1.12 Supported Application Model Package

The "Supported Application Model" package identifies the application models which are supported by the AMSM service.

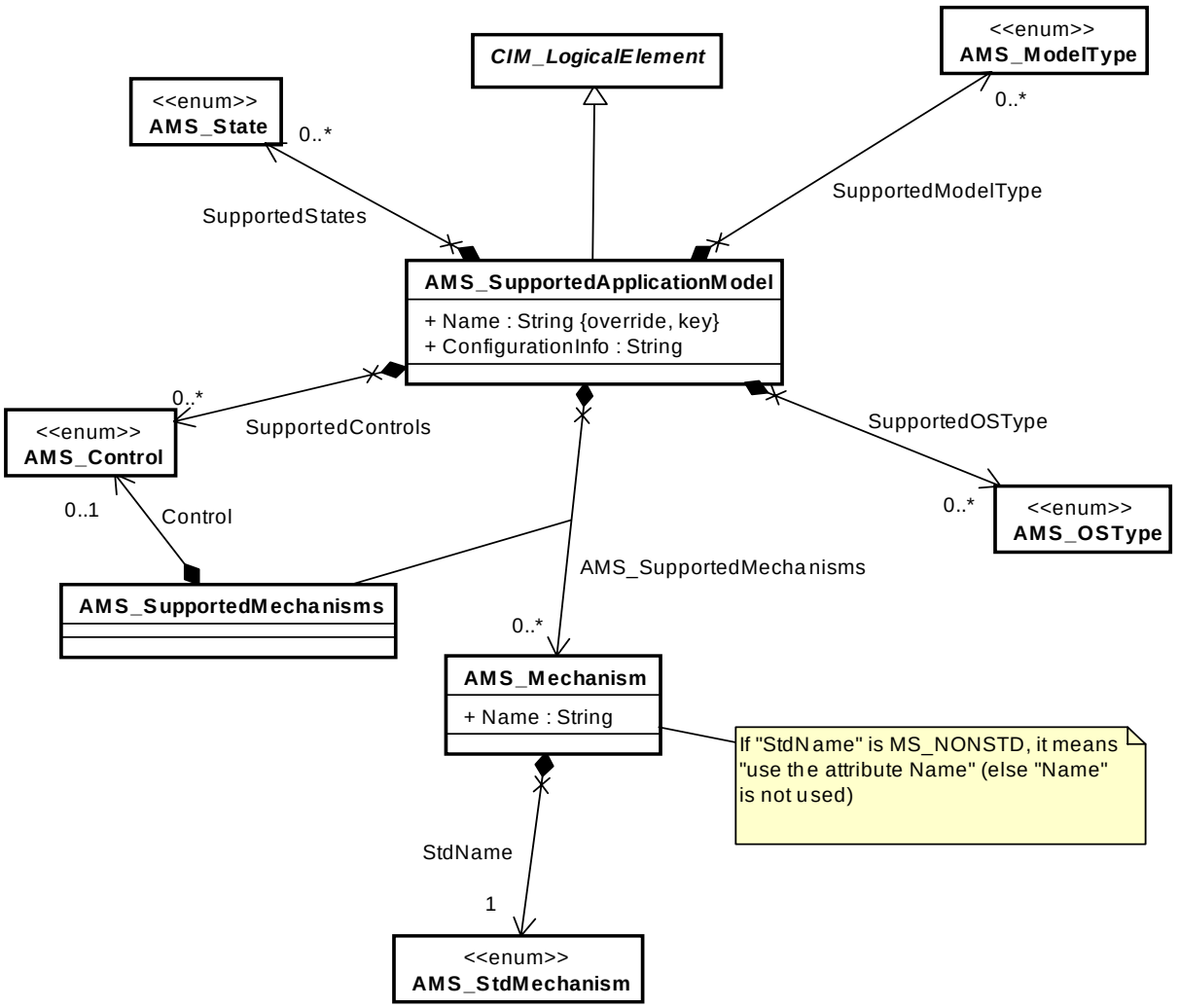


Figure 7.25 - Supported Application Model class diagram

This paragraph defines also the AMS_Property template which design value-name pairs that are used throughout this PIM.

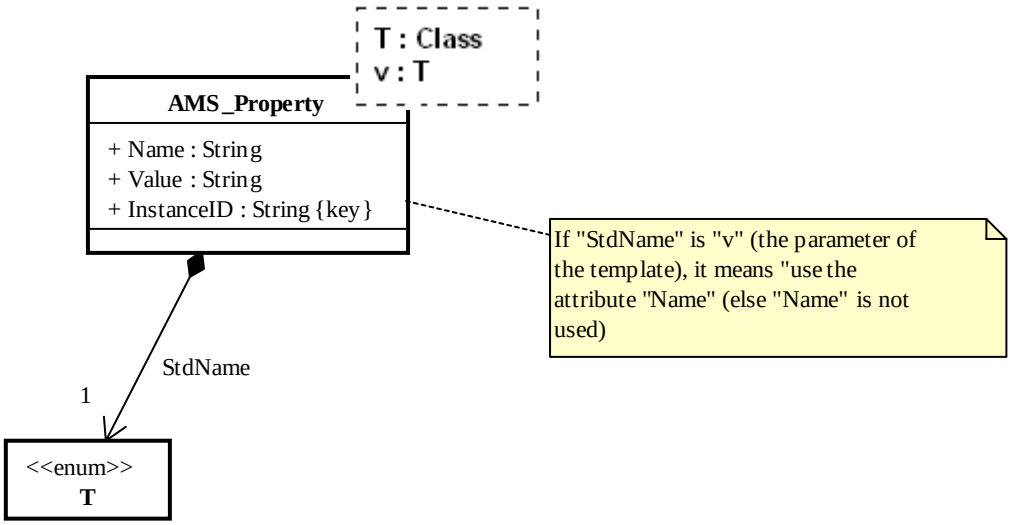


Figure 7.26 - AMS_Property template class diagram

7.1.12.1 AMS_Control Class

This class enumerates the actions allowed when controlling an executable software element. Its items are described in the "Software Element state diagram" (cf. paragraph 2.4.13.).

Some of its items are optional: AMS_LOAD, AMS_START, AMS_UNLOAD, AMS_LOAD_DIRTY ("Maximum Control" profile).

Enumeration
AMS_LOAD
AMS_LOAD_START
AMS_START
AMS_STOP
AMS_HALT
AMS_CONTINUE
AMS_SHUTDOWN
AMS_RECOVER
AMS_UNLOAD
AMS_LOAD_DIRTY
AMS_LOAD_START_DIRTY
AMS_STOP_HALTED

AMS_RECLAIM
AMS_ALLOCATE
AMS_RECOVER_DIRTY

7.1.12.2 AMS_Mechanism Class

The AMS_Mechanism class specifies the possible mechanisms, such as POSIX signals, known by an application model to start, stop, deploy and so forth.

An AMS_Mechanism has a name, defined through a couple string - enumeration (AMS_StdMechanisms): if the value of the enumeration is MS_NONSTD, it means "use the attribute Name" (else "Name" is not used).

Attributes	*Type*	
Name	String	
Associations	*Multiplicity*	*Class*
StdName	1	AMS_StdMechanisms

7.1.12.3 AMS_ModelType Class

This class enumerates the supported types of applications.

Enumeration
AMS_PROCESS
AMS_J2EE
AMS_CCM

7.1.12.4 AMS_OSType Class

This class enumerates supported operating systems.

Enumeration
cf. CIM_OperatingSystem.OSType

7.1.12.5 AMS_Property<Class T, T v> Template

The AMS_Property template aims at designing value-name pairs that are used throughout the PIM. An AMS_Property has a value and a name, which is defined by either a string or an item of an enumeration. This enumeration is intended to normalize some of the possible values of the name yet letting the possibility to the implementations to define new possible names.

The AMS_Property template has two arguments T and v. T is the enumeration and v one the value of the preceding evaluation. "v" is intended to be the special value that denotes a non-normalized name.

Attributes	*Type*	
Name	String	
Value	String	
InstanceID {key}	String	
Associations	*Multiplicity*	*Class*
StdName	1	T

7.1.12.6 AMS_State Class

This class enumerates the states in which executable software elements can be found (cf. section 7.1.14.1).

Some of its items are optional: AMS_LOADED (“Maximum Control” profile).

Enumeration
AMS_EXECUTABLE
AMS_HALTED
AMS_LOADED
AMS_RUNNING
AMS_STOPPED
AMS_UNALLOCATED
AMS_ERROR

7.1.12.7 AMS_StdMechanism Class

This class enumerates the standardized mechanisms to start, stop and deploy. MS_NONSTD is the special value that denotes a non-normalized value and MS_POSIXSIGNAL denotes a mechanism acquainted with POSIX signals.

Enumeration
MS_NONSTD
MS_POSIXSIGNAL

7.1.12.8 AMS_SupportedApplicationModel Class

The AMS_SupportedApplicationModel class specifies the application models which are supported by the AMSM services i.e., applications which AMSM implementation can deal with. Such a model is modeled with:

- The kinds of application concerned (process or J2EE or CCM).

- The kinds of operating systems (enumeration defined in CIM).
- A collection of known control options.
- A collection of known application states.
- A name.
- General configuration information (such as startup command line parameters for starting ORBs).
- A collection of mechanisms for each known control option. These mechanisms specify how to start, stop, deploy and so forth and executable element.

Attributes	*Type*	
Name {override, key}	String	
ConfigurationInfo	String	
Associations	*Multiplicity*	*Class*
SupportedModelType	0 .. *	AMS_ModelType
SupportedOSType	0 .. *	AMS_OSType
SupportedControls	0 .. *	AMS_Control
SupportedStates	0 .. *	AMS_State
AMS_SupportedMechanisms	0 .. *	AMS_Mechanism

7.1.12.9 AMS_SupportedMechanisms Class

The AMS_SupportedMechanisms class is an association class between AMS_SupportedApplicationModel and AMS_Mechanism. It specifies for which type of control a mechanism is supported.

Associations	*Multiplicity*	*Class*
Control	0 .. 1	AMS_Control

7.1.13 Miscellaneous

7.1.13.1 Success or failure codes

Responses to all startup or shutdown requests must return either an indication of success or an indication of failure along with amplifying information concerning the reason for the failure. These indications of success or failure are defined in an enumeration which items are:

Table 7.4 - Error Codes

Enumeration	Comment
AMS_OK	Value is 0. No error.

Table 7.4 - Error Codes

AMS_BADFILTER	A hardware or software filter is badly formed.
AMS_BADSUBSCRIPTIONID	An Unsubscribe operation has been called with an ID which does not (still) exist.
AMS_BADCONNECTIVITY	The ‘connectivity’ parameter passed in the AMS_HWManagement::CreateHardwareGroup method in wrong.
AMS_BADDEVICES	The ‘devices’ parameter passed in the AMS_HWManagement::CreateHardwareGroup method in wrong.
AMS_BADRESOURCES	The ‘resources’ parameter passed in the AMS_HWManagement::CreateHardwareGroup method in wrong.
AMS_BADMODELTYPE	A host does not support the proposed model type.
AMS_BADCOMMANDLINE	A parameter containing a command line is badly formed.
AMS_BADACTION	Badly formed CIM_Action in a specification definition (for example, “CommandLine” and “ProgramPath” of a CIM_ExecuteProgram hold wrong information, CIM_CopyFileAction refers to non-existing directories).
AMS_BADCHECK	Badly formed CIM_Check in a specification definition (for example, an “ArchitectureType” in an CIM_Architecture-Check is unknown, a “TargetOperatingSystem” in an CIM_OSVersionCheck is unknown, the “MinimumVersion” of a CIM_OSVersionCheck if greater than its “MaximumVersion.”
AMS_BADSTATE	One of the action method of the ASM_ExecutableSoftware-Element class is called while the element is not in the proper state for this action according to the state diagram (cf. Section 7.1.14.1).
AMS_STARTFAILED	The Startup method (AMS_SoftwareSystem, AMS_Application, AMS_ExecutableSoftwareElement, AMS_RedundancyGroup, AMS_LoadBalancingGroup, AMS_DeploymentConfiguration) failed for an unknown reason (see log).
AMS_SHUTDOWNFAILED	The Shutdown method (AMS_SoftwareSystem, AMS_Application, AMS_ExecutableSoftwareElement, AMS_RedundancyGroup, AMS_LoadBalancingGroup, AMS_DeploymentConfiguration) failed for an unknown reason (see log).
AMS_LOADFAILED	The AMS_ExecutableSoftwareElement::Load method failed for an unknown reason (see log).
AMS_STOPFAILED	The AMS_ExecutableSoftwareElement::Stop method failed for an unknown reason (see log).
AMS_CONTFAILED	The AMS_ExecutableSoftwareElement::Continue method failed for an unknown reason (see log).
AMS_DEPLOYFAILED	The AMS_DeploymentSpecification::Deploy method failed for an unknown reason (see log).

Table 7.4 - Error Codes

AMS_PRIMARYFAILED	The AMS_ExecutableSoftwareElement::ActivateAsPrimary method failed for an unknown reason (see log).
AMS_RESOURCEERROR	OS resource starvation error (lack of memory, disk).
AMS_RIGHTERROR	OS right error.
AMS_NOTCHECKED	A check of an specification was not true.
AMS_ALREADYPRIMARY	An ActivateAsPrimary has been tried on an element that is already a primary.
AMS_NOTFT	An FT operation has been tried on an element that is not in a redundancy group.

7.1.13.2 Log of activity

The service shall maintain a log of its own activity. The activities recorded shall include at least:

- Status change information (either the stati or else the control commands)
- For failures any amplification of the reason for failure

This log facility is modeled with the classes AMS_Log and AMS_LogRecord. These classes provide a logging facility matching the CIM logging and the Lightweight Logging Service (cf. [LWLOG]).

CIM logging facility is modeled in the CIM package.

Lightweight Logging Service is reused in the “Lightweight Logging Service” package.

7.1.14 Dynamic behavior

7.1.14.1 Software Element state diagram

This State Diagram illustrates the set of States and Transitions that a Software Element may go through. Not all such States are supported by all Software Elements; for instance, the ‘Loaded’ State may be Operating System specific.

The applicable set of States appropriate for a given Software Element, is defined by the SupportedStates attribute of the AMS_SupportedApplicationModel. Also the SupportedControl attribute of the AMS_SupportedApplicationModel indicates which state transitions are available for a given Software Element.

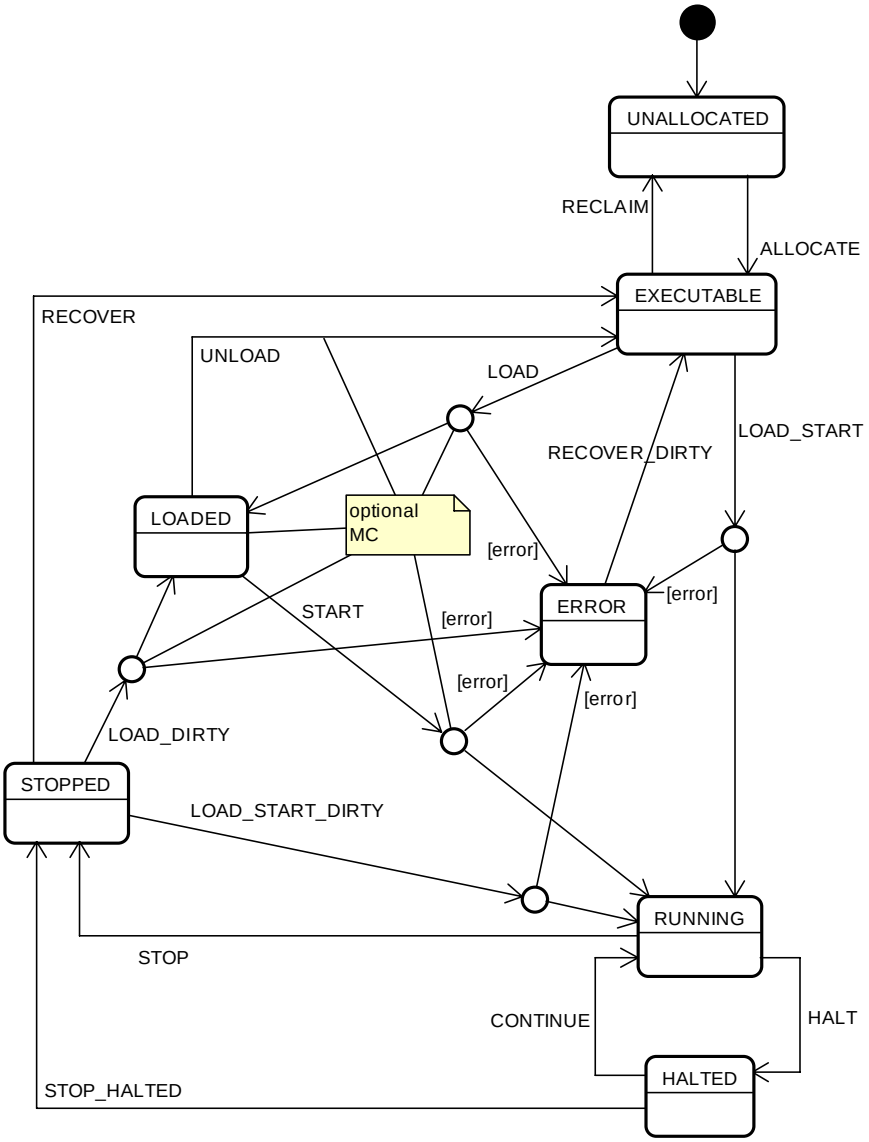


Figure 7.27 - Executable Software Element state diagram

7.1.14.1.1 Unallocated

Description

A Software Element is in the UNALLOCATED state when it has not been allocated to an AMS_ComputerSystem.

Transitions

ALLOCATE	The action of creating an association between this AMS_ExecutableSoftwareElement and an AMS_ComputerSystem.

7.1.14.1.2 Executable

Description

The Software Element is available to be executed; it has been allocated to an AMS_ComputerSystem which has the ability to execute the Software Element. If necessary, the code for the process and associated libraries have been retrieved and are now accessible on the AMS_ComputerSystem.

Transitions

LOAD	Load the Software Element into memory on the associated AMS_ComputerSystem computer. This transition is optional (“Maximum Control” profile).
LOAD_START	Load the Software Element into memory and begin its execution on the associated AMS_ComputerSystem.
RECLAIM	The Software Element is disassociated from the given AMS_ComputerSystem, freeing resources.

7.1.14.1.3 Loaded

Description

The Software Element has been loaded into memory on the associated AMS_ComputerSystem. The Process Counter is set to the first instruction in the process, but execution has not commenced.

This state is optional (“Maximum Control” profile).

Transitions

START	Begin the execution of the Software Element. This transition is optional (“Maximum Control” profile).
UNLOAD	Shutdown the Software Element. This transition is optional (“Maximum Control” profile).

7.1.14.1.4 Running

Description

The Software Element is executing on the associated AMS_ComputerSystem.

Transitions

HALT	Halt the execution of the Software Element (e.g., a SIGSTOP on Unix).
STOP	Stop the Software Element.

7.1.14.1.5 Halted

Description

The Software Element has been Halted. The Process Counter is not advanced while in this state.

Transitions

STOP_HALTED	A request to resume execution of the Software Element (e.g., a SIGCONT on Unix).

7.1.14.1.6 Stopped

Description

The Software Element has finished executing on the given AMS_ComputerSystem.

Transitions

RECOVER	Shutdown the Software Element.
LOAD_DIRTY	Load the Software Element into memory on the associated AMS_ComputerSystem computer. This transition is optional (“Maximum Control” profile).
LOAD_START_DIRTY	Load the Software Element into memory and begin its execution on the associated AMS_ComputerSystem.

7.1.14.1.7 Error

Description

An error has occurred during a transition action of the Software Element.

For example, going from the executable to running state could fail because the LOAD_START action fails, e.g., an exec of a process fails due to memory, incorrect path for the executable, etc.

Transitions

RECOVER_DIRTY	A request to recover from the ERROR state.

7.1.14.2 Mapping between ESE transitions and methods

This paragraph gives the mapping between the methods of AMS_ExecutableSoftwareElement and the transitions known by an executable software element.

Table 7.5 - Mapping between ESE methods and transitions

Methods	Transitions
StartUp, StartUpOnSpec, StartUpOnHost	LOAD_START, START, LOAD_START_DIRTY
ShutDown, ForceShutDown	UNLOAD, RECOVER, STOP, RECOVER_DIRTY
Load	LOAD, LOAD_DIRTY
Stop	HALT, STOP_HALTED
Continue	CONTINUE

8 OMG CORBA/IDL Platform Specific Model

8.1 Mapping Rationale

8.1.1 Objective

The objective of this PSM is to normalize the CORBA/IDL structures and interfaces (cf. [CORBA]).

There are different ways in which this PSM may be utilized; a list (not willing to be exhaustive) is:

- Browsing software system structures: application, groups, and ESEs.
- Browsing networks and computer systems.
- Discovery and configuration of networks and computers.
- Software data inventory.
- Display of computers and/or applications statuses.
- GUI-based management of applications and computers.

These uses may be gathered in two main purposes: firstly getting information from a database of software and hardware, secondly managing some of these element: applications, ESEs, computers. If the first purpose involves interfaces to get all the attributes and to iterate on all the associations, the later one asks for a way to quickly retrieve elements of the object.

Therefore all attributes, methods and associations are mapped into IDL elements. As a general rule, classes are mapped to interfaces, attributes to CORBA/IDL attributes, associations to read-only attributes and methods to equivalent methods which deal with errors through CORBA exceptions. Some classes (indications, filters, association classes) are not considered as interface *per se* but as pure information and, so, they are mapped to CORBA/IDL structs.

CORBA/IDL datatypes have not been considered since firstly no method clearly requires to be ran on the clients, and secondly datatypes implies to supply “server libraries” for each possible platform (Java, C++) with .idl files.

Moreover, all associations are mapped with an additional iterator on the list of elements of this association and all methods which return a list of objects are also mapped with an additional method which returns such an iterator.

This iterator is an interface which implements the well-known “Iterator” design pattern (cf. “Design Patterns – Elements of Reusable Object-Oriented Software” Gamma, Helm, Johnson and Vissides – Addison-Wesley 1994) on the associated elements gathered in a list. This interface defines the methods “First,” “Next,” “IsDone,” and “GetCurrentItem.” The method “First” make the iterator stand on the first element of the list, “Next” shifts it on the following element, “IsDone” returns “TRUE” if the iterator is out of bound and “GetCurrentItem” returns the element currently referenced by the iterator.

Subscribe methods and Indication classes are also mapped with one client IDL which has to be implemented by clients in order to receive indications (i.e., callbacks) from an AMSM server.

8.1.2 Mapping principle

The rules used to map the PIM into CORBA/IDL are:

- All package contents are mapped in a single .idl file in which the prefix “omg.org” is used in order to cleanly organize name scope in the interface repository:

```
// Copyright 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
// Themis Computer and Progeny Systems Corporation.

#ifndef _AMS_AMSManagement_IDL_
#define _AMS_AMSManagement_IDL_

#pragma prefix "omg.org"

#include "AMS_Util.idl"
...
#endif/* _AMS_AMSManagement_IDL_ */
```

- All packages are mapped to a module which gathers all the definitions of the package

```
module AMS_Application {
   ...
};
```

- Data types (uint16, datetime…) are directly mapped to CORBA/IDL types using the rules explained in paragraph 2.5.3.
- Each enumeration class is mapped to a corresponding CORBA/IDL enumeration type and to the type of sequence referred to this enumeration as well:

```
enum enumeration-name {
   an-item,
   ...
};
typedef sequence<enumeration-type> enumeration-typeList;
```

- Non enumeration classes are pre-declared as an interface with:
- A constant which contains the repository identifier of this interface. This constant is intended to be used with the “Object::is_a” method.
- The type of the sequence of this interface,
- An interface which defines the type of iterator of elements belonging to the class.

```
const string AMS_class-name_CLASSID
           = "IDL:omg.org/package-name/class-name:1.0";
interface class-name ;
typedef sequence< class-name > class-nameList;
interface class-nameListIterator : AMS_Util::AMS_Iterator {
   class-name getCurrentItem() raises (AMS_Util::AMS_NoSuchElementException);
};
```

- Non enumeration classes are mapped to interfaces whose name is the name of the class and the inheritance list is the one designed in the PIM.

```
interface class-name : super-class-name {
   ...
};
```

- AMS_Property<C,V> template is mapped as follows:
- An interface named AMS_Property followed by an "_" and the name that is used for the first parameter in the template

spec. (without "AMS"); this interface will contain an attribute whose type is the type of the enumeration coming from the first parameter of the template and whose name is "StdName:"

```
interface AMS_Property_C {
   ...
   attribute V StdName;
};
```

- Attributes of non-enumeration classes are next mapped with a CORBA/IDL attribute whose name is the name of the attribute.

  `attribute` *`attribute-type-map attribute-name`*`;`

- Associations of non-enumeration classes are next mapped as follow:
- Compositions with multiplicity equal to 1 are mapped to an attribute whose name is the name of the association without the leading CIM_ or ASM_ and whose type is the type of the target class of the composition (which is, in the AMSM PIM, always an enumeration class).

  `attribute class-name` *`assoc-name`*`;`

- Compositions with multiplicity superior to 1 are mapped to an attribute whose name is the name of the association without the leading CIM_ or ASM_ and type is a sequence of elements with the type of the target class of the composition (which is, in the AMSM PIM, always an enumeration class).

  `attribute class-nameList` *`assoc-name`*`;`

- Associations whose reverse association is a composition are not mapped.
- Associations whose reverse association is an aggregation are mapped with a read-only attribute whose name is “Owner” and type the target of the association:

  `readonly attribute class-name Owner;`

- Remaining associations are mapped to a read-only attribute whose name is the name of the association without the leading CIM_ or ASM_ and whose type is the target of the association if the corresponding maximum multiplicity is 1, and a sequence of the target of the association otherwise. In the latter case, an operation which returns an iterator on the association is also given.
- Case maximum multiplicity = 1:

  `readonly attribute` *`assoc-type assoc-name`*`;`

- Else:

  `readonly attribute` *`assoc-type`*`List` *`assoc-name`*`;`
  `assoc-type``ListIterator Get`*`assoc-type`*`Iterator ( );`

- Methods are mapped to CORBA/IDL methods with the same parameters. The return type is void and the mapped method may raise an exception of type AMS_Error which contains the AMSM code of the error.

Those rules are not intended to be general rules to map an UML PIM to a CORBA/IDL file. They are specific rules established for the AMSM specific case.

8.1.3 Mapping exceptions

The preceding rules deal neither with all the cases which arise in the AMSM PIM, nor with some of the requirements of CORBA/IDL norm.

Hereby are those specific cases:

- File and module AMS_Util is added to declare some PSM-wide used definitions (exceptions, error codes, iterators).
- File and module AMS_Client is added to declare the interface that clients have to implement in order to receive callbacks from an AMSM server. This interface gets two one way methods: one for hardware indications, the other one for software indications.
- Non-alphabetical and non-numerical characters in enumeration items (such as '/', '(', ')') are mapped to underscores.
- Module AMS_SupportedApplicationModel is renamed with AMS_SAM in order to avoid name mismatch with the interface AMS_SupportedApplicationModel.
- Module AMS_Application is renamed with AMS_ApplicationModule in order to avoid name mismatch with the interface AMS_Application.
- Some associations such as CIM_SystemComponent or CIM_RedundancyComponent and attributes (including the special case "Owner") appears two or three times on a same class, which leads to multiple definition. Since this is not allowed in a CORBA/IDL, those multiple definitions are gathered in one of type "Object." The client will have next to test this "Object" against all possible types of the association by using the method Object::is_a and predefined repository identifier constants.
- Classes which are association classes are mapped to structures.
- Associations which are association classes are mapped with additional methods which return the list of elements or an iterator on this list, and take as parameter the structure corresponding to the association class.
- Classes which are the target of an association class do not map this association as usually but replace it with an attribute whose name is "State" and type is the structure of the association class.
- AMS_RTHWIndication, AMS_RTSWIndication, AMS_SWFilter and AMS_HWFilter are mapped to structures.
- Methods which return collection of elements on Management classes are mapped with another method which returns the corresponding iterator.
- Subscribe methods take an additional (first) parameter holding the CORBA object which will be called back.

8.1.4 Initial reference issues

A client application which wants to contact the AMSM service have to create or get a proxy. A simple way to allow it is to reference the AMSM service objects in the Naming Service.

If the AMSM service is referenced in the Naming Service, the names and context which must be used are given by the following tables:

Issue 11406 - Incorrect numbering of tables

Table 8.1 - AMSM entry points in the Naming Service

Interface (in the AMS_AMSManagement module)	Context:Name	Comment
AMS_HWManagement	AMSM:HWManagement	To get computers, end points, networks, domains, and hardware groups
AMS_DeploymentConfManagement	AMSM:DeploymentMgt	To get deployment configurations
AMS_SystemManagement	AMSM:SystemMgt	To get software systems
AMS_ApplicationManagement	AMSM:ApplicationMgt	To get applications
AMS_RedundancyGroupManagement	AMSM:FTGroupMgt	To get redundancy groups
AMS_LoadBalancingManagement	AMSM:LBGroupMgt	To get load balancing groups
AMS_ESEManagement	AMSM:ESEMgt	To get executable software elements
AMS_SAMManagement	AMSM:SAMMgt	To get supported application models
AMS_ConfManagement	AMSM:CongMgt	To load and unload a configuration
AMS_Log	AMSM:Log	To get the log

8.2 Specific Attributes and Parameters Information

In this paragraph, some attributes of class and parameters of operation roughly defined in the PIM, are more specified in the context of the CORBA/IDL PSM.

Issue 11406 - Incorrect numbering of tables

Table 8.2 - Specific attributes for CORBA/IDL PSM

Attribute	Comment
AMS_HWFilter:filter	Implementation dependent
AMS_SWFilter:filter	Implementation dependent
AMS_SupportedApplicationModel: ConfigurationInfo	Implementation dependent
AMS_ExecuteProgram:Environment	Implementation dependent
AMS_ExecuteProgram:CommandLine	Implementation dependent
AMS_ExecuteProgram:ProgramPath	Implementation dependent

Issue 11406 - Incorrect numbering of tables

Table 8.3 - Specific parameters for CORBA/IDL PSM

Parameter	Comment
CreateHardwareGroup: connectivity	Implementation dependent
CreateHardwareGroup: devices	Implementation dependent
CreateHardwareGroup: resources	Implementation dependent

8.3 Specific Data Types

Issue 11406 - Incorrect numbering of tables

This paragraph specifies the data types in the context of the CORBA/IDL PSM.

Table 8.4 - Data Types for CORBA/IDL PSM

Data type	Definition	Comment
Collection<Type>	`typedef sequence<class-name> class-nameList;`	If Type is either an enumeration or a structure

Collection<Type>	`typedef sequence<class-name> class-nameList;`
	`interface class-nameListIterator :`
	`AMS_Util::AMS_Iterator {`
	`  class-name getCurrentItem()`
	`  raises (AMS_NoSuchElementException);`
	`  };`
		If Type is neither an enumeration nor a structure.
		The Iterator interface implements the well-known “Iterator” design pattern.

datetime	`typedef string AMS_datetime;`	
String	string	
uint8	`typedef unsigned shortAMS_uint8;`	
uint16	`typedef unsigned shortAMS_uint16;`	
uint32	`typedef unsigned longAMS_uint32;`	
uint64	`typedef unsigned long longAMS_uint64;`	

8.4 Specific Failure Codes

Error codes have been defined in the PIM (cf. 7.1.13.1).

8.5 Conformance Criteria

This PSM acknowledges the same conformance criteria as the PIM (cf. Chapter 2). So, classes, attributes, and methods which are not known in a PIM compliance profile are not mapped to the corresponding PSM compliance profile.

8.6 Mapping

8.6.1 AMS_Util.idl

```
// Copyright 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
// Themis Computer and Progeny Systems Corporation.

#include "orb.idl"

#ifndef _AMS_Util_IDL_
#define _AMS_Util_IDL_

#pragma prefix "omg.org"

module AMS_Util {

   typedef string AMS_datetime;
   typedef unsigned shortAMS_uint8;
   typedef unsigned shortAMS_uint16;
   typedef unsigned longAMS_uint32;
   typedef unsigned long longAMS_uint64;

    exception AMS_NoSuchElementException {
   };

    enum AMS_ErrorCode {
      AMS_UNKNOWN,
      AMS_BADFILTER,
      AMS_BADSUBSCRIPTIONID,
      AMS_BADCONNECTIVITY,
      AMS_BADDEVICES,
      AMS_BADRESOURCES,
      AMS_BADMODELTYPE,
      AMS_BADCOMMANDLINE,
      AMS_BADACTION,
      AMS_BADCHECK,
      AMS_BADSTATE,
      AMS_STARTFAILED,
      AMS_SHUTDOWNFAILED,
      AMS_LOADFAILED,
      AMS_STOPFAILED,
      AMS_CONTFAILED,
      AMS_DEPLOYFAILED,
      AMS_PRIMARYFAILED,
      AMS_RESOURCEERROR,
      AMS_RIGHTERROR,
      AMS_NOTCHECKED,
      AMS_ALREADYPRIMARY,
      AMS_NOTFT
    };

    exception AMS_Error {
       AMS_ErrorCode error_code;
```

```
    };

    interface AMS_Iterator {
      void First ();
      void Next ();
      // XXX GetCurrentItem () raises (AMS_NoSuchElementException);
      boolean IsDone ();
    };

   typedef sequence<Object> AMS_ObjectList;
   interface AMS_ObjectListIterator : AMS_Iterator {
      AMS_ObjectList GetCurrentItem ()
          raises (AMS_NoSuchElementException);
   };

   typedef sequence<string> AMS_stringList;

};


#endif/* _AMS_Util_IDL_ */
```

8.6.2 AMS_Client.idl

```
// Copyright 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
// Themis Computer and Progeny Systems Corporation.

#include "orb.idl"

#ifndef _AMS_Client_IDL_
#define _AMS_Client_IDL_

#pragma prefix "omg.org"

#include "AMS_AMSManagement.idl"

module AMS_ClientModule {

   interface AMS_IndicationSink {
      oneway void notifyHW ( in AMS_AMSManagement::AMS_RTHWIndication indic);
      oneway void notifySW ( in AMS_AMSManagement::AMS_RTSWIndication indic);
   };
};

#endif/* _AMS_Client_IDL_ */
```

8.6.3 AMS_AMSManagement.idl

```
// Copyright 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
// Themis Computer and Progeny Systems Corporation.

#include "AMS_Util.idl"
#include "AMS_CIM.idl"

#ifndef _AMS_AMSManagement_IDL_
#define _AMS_AMSManagement_IDL_

#pragma prefix "omg.org"
```

```
#include "AMS_Application.idl"
#include "AMS_ApplicationDeployment.idl"
#include "AMS_ApplicationDeploymentSpecification.idl"
#include "AMS_ApplicationSpecification.idl"
#include "AMS_LogicalHardware.idl"
#include "AMS_LogicalHardwareSpecification.idl"
#include "AMS_SupportedApplicationModel.idl"
#include "AMS_LightweightLoggingService.idl"


module AMS_Client {
   interface AMS_IndicationSink;
} ;

module AMS_AMSManagement {
   struct AMS_ErrorStruct {
      string Message;
      AMS_Util::uint16 Number;
      AMS_Util::AMS_ErrorCode Code;
      string Element;
   };

   //
   const string AMS_HWManagement_CLASSID
         = "IDL:omg.org/AMS_AMSManagement/AMS_HWManagement:1.0";
   interface AMS_HWManagement ;

   //
   const string AMS_DeploymentConfManagement_CLASSID
         = "IDL:omg.org/AMS_AMSManagement/AMS_DeploymentConfManagement:1.0";
   interface AMS_DeploymentConfManagement ;

   //
   const string AMS_ESEManagement_CLASSID
         = "IDL:omg.org/AMS_AMSManagement/AMS_ESEManagement:1.0";
   interface AMS_ESEManagement ;

   //
   const string AMS_ApplicationManagement_CLASSID
         = "IDL:omg.org/AMS_AMSManagement/AMS_ApplicationManagement:1.0";
   interface AMS_ApplicationManagement ;

   //
   const string AMS_SystemManagement_CLASSID
         = "IDL:omg.org/AMS_AMSManagement/AMS_SystemManagement:1.0";
   interface AMS_SystemManagement ;

   //
   const string AMS_RedundancyGroupManagement_CLASSID
         = "IDL:omg.org/AMS_AMSManagement/AMS_RedundancyGroupManagement:1.0";
   interface AMS_RedundancyGroupManagement ;

   //
   const string AMS_LoadBalancingManagement_CLASSID
         = "IDL:omg.org/AMS_AMSManagement/AMS_LoadBalancingManagement:1.0";
   interface AMS_LoadBalancingManagement ;

   //
   const string AMS_ConfManagement_CLASSID
         = "IDL:omg.org/AMS_AMSManagement/AMS_ConfManagement:1.0";
```

```
interface AMS_ConfManagement ;

//
const string AMS_Log_CLASSID
      = "IDL:omg.org/AMS_AMSManagement/AMS_Log:1.0";
interface AMS_Log ;
typedef sequence<AMS_Log> AMS_LogList;
interface AMS_LogListIterator : AMS_Util::AMS_Iterator {
   AMS_Log GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_LogRecord_CLASSID
      = "IDL:omg.org/AMS_AMSManagement/AMS_LogRecord:1.0";
interface AMS_LogRecord ;
typedef sequence<AMS_LogRecord> AMS_LogRecordList;
interface AMS_LogRecordListIterator : AMS_Util::AMS_Iterator {
   AMS_LogRecord GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
struct AMS_HWFilter {
   AMS_LogicalHardware::AMS_HardwareGroupList GroupsInFilter;
   AMS_LogicalHardware::AMS_DomainList DomainsInFilter;
   AMS_CIM::CIM_NetworkList NetworkInFilter;
   string filter;
};

//
struct AMS_SWFilter {
   AMS_SAM::AMS_StateList StateFilters;
   string filter;
};

//
struct AMS_RTHWIndication {
   // when
   AMS_Util::AMS_datetime IndicationTime;
   // who
   AMS_LogicalHardware::AMS_ComputerSystem NetworkElt;
   // status
   AMS_LogicalHardware::AMS_Property_StdHWUtilisation RTHS;
};
typedef sequence<AMS_RTHWIndication> AMS_RTHWIndicationList;

//
struct AMS_RTSWIndication {
   // when
   AMS_Util::AMS_datetime IndicationTime;
   // who (one of)
   AMS_ApplicationModule::AMS_ExecutableSoftwareElement ESE;
   AMS_ApplicationModule::AMS_Application AppIndication;
   AMS_ApplicationModule::AMS_SoftwareSystem System;
   AMS_ApplicationModule::AMS_LoadBalancingGroup LB;
   AMS_ApplicationModule::AMS_RedundancyGroup RG;
   // status
   AMS_ApplicationModule::AMS_RTSoftwareStatus RTSW;
};
```

```
typedef sequence<AMS_RTSWIndication> AMS_RTSWIndicationList;

//
interface AMS_HWManagement {
   AMS_CIM::CIM_ProtocolEndPointList GetNetworkLinks ( in AMS_HWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_CIM::CIM_ProtocolEndPointListIterator GetNetworkLinksIterator (
                                          in AMS_HWFilter filter )
      raises (AMS_Util::AMS_Error);
   void SubscribeNetworkLoadChange ( in AMS_Client::AMS_IndicationSink sink,
                           in AMS_HWFilter filter,
                           out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void SubscribeNetworkLoad ( in AMS_Client::AMS_IndicationSink sink,
                     in AMS_Util::AMS_uint16 delay,
                     in AMS_HWFilter filter,
                     out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void CreateHardwareGroup ( in AMS_CIM::CIM_Location location,
                     in string connectivity,
                     in string devices,
                     out AMS_LogicalHardware::AMS_HardwareGroup group )
      raises (AMS_Util::AMS_Error);
   AMS_CIM::CIM_LocationList GetAllLocations ( )
      raises (AMS_Util::AMS_Error);
   AMS_CIM::CIM_LocationListIterator GetAllLocationsIterator ( )
      raises (AMS_Util::AMS_Error);
   void SubscribeHWStatusChange ( in AMS_Client::AMS_IndicationSink sink,
                     in AMS_HWFilter filter,
                     out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void SubscribeHWStatus ( in AMS_Client::AMS_IndicationSink sink,
                     in AMS_Util::AMS_uint16 delay,
                     in AMS_HWFilter filter,
                     out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void Unsubscribe ( in AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   AMS_LogicalHardware::AMS_ComputerSystemList GetComputerSystems (
                                          in AMS_HWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_LogicalHardware::AMS_ComputerSystemListIterator GetComputerSystemsIterator (
                                          in AMS_HWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_LogicalHardware::AMS_HardwareGroupList GetHardwareGroups (
                                          in AMS_HWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_LogicalHardware::AMS_HardwareGroupListIterator GetHardwareGroupsIterator (
                                          in AMS_HWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_CIM::CIM_NetworkList GetNetworks ( in AMS_HWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_CIM::CIM_NetworkListIterator GetNetworksIterator ( in AMS_HWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_LogicalHardware::AMS_DomainList GetDomains ( in AMS_HWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_LogicalHardware::AMS_DomainListIterator GetDomainsIterator (
                                          in AMS_HWFilter filter )
      raises (AMS_Util::AMS_Error);
};
```

```
//
interface AMS_DeploymentConfManagement {
   AMS_ApplicationDeployment::AMS_DeploymentConfigurationList
                                          GetDeploymentConfiguration ( )
      raises (AMS_Util::AMS_Error);
   AMS_ApplicationDeployment::AMS_DeploymentConfigurationListIterator
                                  GetDeploymentConfigurationIterator ( )
      raises (AMS_Util::AMS_Error);
};

//
interface AMS_ESEManagement {
   AMS_ApplicationModule::AMS_ExecutableSoftwareElementList GetESE (
                                          in AMS_SWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_ApplicationModule::AMS_ExecutableSoftwareElementListIterator GetESEIterator (

                                          in AMS_SWFilter filter )
      raises (AMS_Util::AMS_Error);
   void SubscribeESEStatusChange ( in AMS_Client::AMS_IndicationSink sink,
                        in AMS_SWFilter filter,
                        out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void SubscribeESEStatus ( in AMS_Client::AMS_IndicationSink sink,
                     in AMS_Util::AMS_uint16 delay,
                     in AMS_SWFilter filter,
                     out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void ShutDownESE ( in AMS_SWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_ApplicationSpecification::AMS_ESESpecList GetESESpec ( )
      raises (AMS_Util::AMS_Error);
   AMS_ApplicationSpecification::AMS_ESESpecListIterator GetESESpecIterator ( )
      raises (AMS_Util::AMS_Error);
   void Unsubscribe ( in AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
};

//
interface AMS_ApplicationManagement {
   AMS_ApplicationModule::AMS_ApplicationList GetApplication (
                                          in AMS_SWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_ApplicationModule::AMS_ApplicationListIterator
                        GetApplicationIterator ( in AMS_SWFilter filter )
      raises (AMS_Util::AMS_Error);
   void SubscribeApplicationStatusChange ( in AMS_Client::AMS_IndicationSink sink,
                              in AMS_SWFilter filter,
                              out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void SubscribeApplicationStatus ( in AMS_Client::AMS_IndicationSink sink,
                        in AMS_Util::AMS_uint16 delay,
                        in AMS_SWFilter filter,
                        out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void Unsubscribe ( in AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
};
```

```
//
interface AMS_SystemManagement {
   AMS_ApplicationModule::AMS_SoftwareSystemList GetSystem (
                                          in AMS_SWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_ApplicationModule::AMS_SoftwareSystemListIterator GetSystemIterator(
                                          in AMS_SWFilter filter )
      raises (AMS_Util::AMS_Error);
   void SubscribeSystemStatusChange ( in AMS_Client::AMS_IndicationSink sink,
                        in AMS_SWFilter filter,
                        out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void SubscribeSystemStatus ( in AMS_Client::AMS_IndicationSink sink,
                        in AMS_Util::AMS_uint16 delay,
                        in AMS_SWFilter filter,
                        out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void Unsubscribe ( in AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
};

//
interface AMS_RedundancyGroupManagement {
   AMS_ApplicationModule::AMS_RedundancyGroupList GetRG ( in AMS_SWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_ApplicationModule::AMS_RedundancyGroupListIterator GetRGIterator (
                                             in AMS_SWFilter filter )
      raises (AMS_Util::AMS_Error);
   void SubscribeRGStatusChange ( in AMS_Client::AMS_IndicationSink sink,
                        in AMS_SWFilter filter,
                        out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void SubscribeRGStatus ( in AMS_Client::AMS_IndicationSink sink,
                     in AMS_Util::AMS_uint16 delay,
                     in AMS_SWFilter filter,
                     out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void Unsubscribe ( in AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
};

//
interface AMS_LoadBalancingManagement {
   AMS_ApplicationModule::AMS_LoadBalancingGroupList GetLB (
                                             in AMS_SWFilter filter )
      raises (AMS_Util::AMS_Error);
   AMS_ApplicationModule::AMS_LoadBalancingGroupListIterator GetLBIterator(
                                             in AMS_SWFilter filter )
      raises (AMS_Util::AMS_Error);
   void SubscribeLBStatusChange ( in AMS_Client::AMS_IndicationSink sink,
                     in AMS_SWFilter filter,
                     out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void SubscribeLBStatus ( in AMS_Client::AMS_IndicationSink sink,
                  in AMS_Util::AMS_uint16 delay,
                  in AMS_SWFilter filter,
                  out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void Unsubscribe ( in AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
```

```
   };

   //
   interface AMS_ConfManagement {
      void LoadConfiguration(in string file)
         raises (AMS_Util::AMS_Error);
      void UnloadConfiguration(in string file)
         raises (AMS_Util::AMS_Error);
      AMS_ErrorStruct GetLastError ()
         raises (AMS_Util::AMS_Error);
   };

   //
   interface AMS_SAMManagement {
      AMS_SAM::AMS_SupportedApplicationModelList GetAllSAM ( )
         raises (AMS_Util::AMS_Error);
      AMS_SAM::AMS_SupportedApplicationModelListIterator GetAllSAMIterator ( )
         raises (AMS_Util::AMS_Error);
   };

   //
   interface AMS_Log : AMS_CIM::CIM_Log {
      AMS_LogRecordListIterator GetLogManagesRecordIterator ( );
      readonly attribute AMS_LogRecordList LogManagesRecord;
      readonly attribute AMS_LightweightLoggingService::LogAdministrator
                                                          LogAdministrator;
      readonly attribute AMS_LightweightLoggingService::LogConsumer LogConsumer;
   };

   //
   interface AMS_LogRecord : AMS_CIM::CIM_RecordForLog {
      readonly attribute AMS_Log Owner;
      readonly attribute AMS_LightweightLoggingService::LogRecord LogicalIdentity;
   };

};

#endif/* _AMS_AMSManagement_IDL_ */
```

8.6.4 AMS_Application.idl

```
// Copyright 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
// Themis Computer and Progeny Systems Corporation.

#include "AMS_Util.idl"
#include "AMS_CIM.idl"

#ifndef _AMS_Application_IDL_
#define _AMS_Application_IDL_

#pragma prefix "omg.org"

#include "AMS_SupportedApplicationModel.idl"
#include "AMS_ApplicationSpecification.idl"
#include "AMS_ApplicationDeployment.idl"

module AMS_ApplicationModule {
   //
   enum AMS_ReplicationStyle {
      RG_COLD_PASSIVE,
```

```
   RG_WARM_PASSIVE,
   RG_ACTIVE,
   RG_ACTIVE_WITH_VOTING,
   RG_STATELESS,
   RG_IMPL_DEFINED
};
typedef sequence<AMS_ReplicationStyle> AMS_ReplicationStyleList;
//
enum AMS_BalancingStyle {
   LB_ROUND_ROBIN,
   LB_RANDOM,
   LB_IMPL_DEFINED
};
typedef sequence<AMS_BalancingStyle> AMS_BalancingStyleList;
//
enum AMS_RTSoftwareStatus {
   SW_STARTED,
   SW_STOPPED,
   SW_FAILED
};
typedef sequence<AMS_RTSoftwareStatus> AMS_RTSoftwareStatusList;
//
enum AMS_StdState {
   ST_NONSTD,
   ST_ENV
};
typedef sequence<AMS_StdState> AMS_StdStateList;
//
const string AMS_SoftwareSystem_CLASSID
      = "IDL:omg.org/AMS_Application/AMS_SoftwareSystem:1.0";
interface AMS_SoftwareSystem ;
typedef sequence<AMS_SoftwareSystem> AMS_SoftwareSystemList;
interface AMS_SoftwareSystemListIterator : AMS_Util::AMS_Iterator {
   AMS_SoftwareSystem GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_Application_CLASSID
      = "IDL:omg.org/AMS_Application/AMS_Application:1.0";
interface AMS_Application ;
typedef sequence<AMS_Application> AMS_ApplicationList;
interface AMS_ApplicationListIterator : AMS_Util::AMS_Iterator {
   AMS_Application GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_ExecutableSoftwareElement_CLASSID
      = "IDL:omg.org/AMS_Application/AMS_ExecutableSoftwareElement:1.0";
interface AMS_ExecutableSoftwareElement ;
typedef sequence<AMS_ExecutableSoftwareElement> AMS_ExecutableSoftwareElementList;
interface AMS_ExecutableSoftwareElementListIterator : AMS_Util::AMS_Iterator {
   AMS_ExecutableSoftwareElement GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_RedundancyGroup_CLASSID
      = "IDL:omg.org/AMS_Application/AMS_RedundancyGroup:1.0";
```

```
interface AMS_RedundancyGroup ;
typedef sequence<AMS_RedundancyGroup> AMS_RedundancyGroupList;
interface AMS_RedundancyGroupListIterator : AMS_Util::AMS_Iterator {
   AMS_RedundancyGroup GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_LoadBalancingGroup_CLASSID
      = "IDL:omg.org/AMS_Application/AMS_LoadBalancingGroup:1.0";
interface AMS_LoadBalancingGroup ;
typedef sequence<AMS_LoadBalancingGroup> AMS_LoadBalancingGroupList;
interface AMS_LoadBalancingGroupListIterator : AMS_Util::AMS_Iterator {
   AMS_LoadBalancingGroup GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_Property_StdState_CLASSID
      = "IDL:omg.org/AMS_Application/AMS_Property_StdState:1.0";
interface AMS_Property_StdState;
typedef sequence<AMS_Property_StdState> AMS_Property_StdStateList;
interface AMS_Property_StdStateListIterator : AMS_Util::AMS_Iterator {
   AMS_Property_StdState GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
interface AMS_Property_StdState {
   attribute string Name;
   attribute string Value;
   attribute string InstanceID;
   attribute AMS_StdState Status;
};

//
interface AMS_SoftwareSystem : AMS_CIM::CIM_ApplicationSystem {
   // list of AMS_Application, AMS_SoftwareSystem or AMS_ExecutableSoftwareElement
   AMS_Util::AMS_ObjectListIterator GetSystemComponentIterator ( );
   readonly attribute AMS_Util::AMS_ObjectList SystemComponent;
   //
   readonly attribute AMS_SoftwareSystem Owner;
   readonly attribute AMS_ApplicationSpecification::AMS_SoftwareFeatureSpec
                                                      SystemFeature;
   attribute AMS_RTSoftwareStatus Status;
   attribute string Name;
   void StartUp ( )
      raises (AMS_Util::AMS_Error);
   void ShutDown ( )
      raises (AMS_Util::AMS_Error);
};

//
interface AMS_Application : AMS_CIM::CIM_ApplicationSystem {
   // list of AMS_ExecutableSoftwareElement, AMS_RedundancyGroup
   // or AMS_LoadBalancingGroup
   AMS_Util::AMS_ObjectListIterator GetSystemComponentIterator ( );
   readonly attribute AMS_Util::AMS_ObjectList SystemComponent;
   // AMS_SoftwareSystem or AMS_Application
   readonly attribute Object Owner;
```

```
      //
      readonly attribute AMS_ApplicationSpecification::AMS_SoftwareFeatureSpec
                                                          ApplicationFeature;
      AMS_ApplicationListIterator GetApplicationOfApplicationIterator ( );
      readonly attribute AMS_ApplicationList ApplicationOfApplication;
      attribute AMS_RTSoftwareStatus Status;
      attribute string Name;
      void StartUp ( )
         raises (AMS_Util::AMS_Error);
      void ShutDown ( )
         raises (AMS_Util::AMS_Error);
   };

   //
   interface AMS_ExecutableSoftwareElement : AMS_CIM::CIM_Service {
      // AMS_Application or AMS_SoftwareSystem
      readonly attribute Object Owner;
      //
      readonly attribute AMS_ApplicationSpecification::AMS_ESESpec
                                  SoftwareElementServiceImplementation;
      attribute AMS_SAM::AMS_State CurrentState;
      // AMS_RedundancyGroup or AMS_LoadBalancingGroup
      readonly attribute Object RedundancyComponent;
      //
      attribute AMS_RTSoftwareStatus Status;
      attribute AMS_ApplicationSpecification::AMS_RedundancyEltState RedState;
      readonly attribute AMS_ApplicationDeployment::AMS_DeploymentLink ESEDeployed;
      attribute string Name;
      void StartUp ( )
         raises (AMS_Util::AMS_Error);
      void StartUpOnSpec ( in AMS_ApplicationSpecification::AMS_ESESpec spec )
         raises (AMS_Util::AMS_Error);
      void StartUpOnHost(in AMS_LogicalHardware::AMS_Host host, in string commandLine )
         raises (AMS_Util::AMS_Error);
      void ShutDown ( )
         raises (AMS_Util::AMS_Error);
      void Load ( )
         raises (AMS_Util::AMS_Error);
      void LoadAndStart ( )
         raises (AMS_Util::AMS_Error);
      void Stop ( )
         raises (AMS_Util::AMS_Error);
      void Continue ( )
         raises (AMS_Util::AMS_Error);
      void ForceShutDown ( )
         raises (AMS_Util::AMS_Error);
      void ActivateAsPrimary ( )
         raises (AMS_Util::AMS_Error);
      //
      AMS_Property_StdStateListIterator GetStdStateListIterator ( );
      readonly attribute AMS_Property_StdStateList AMS_SpecificState;
   };

   //
   interface AMS_RedundancyGroup : AMS_CIM::CIM_RedundancyGroup {
      AMS_ExecutableSoftwareElementListIterator GetRedundancyComponentIterator ( );
      readonly attribute AMS_ExecutableSoftwareElementList RedundancyComponent;
      readonly attribute AMS_Application Owner;
      readonly attribute AMS_ApplicationSpecification::AMS_SoftwareFeatureSpec
                                                          RedundancyFeature;
```

```
      attribute AMS_ReplicationStyle Style;
      attribute AMS_RTSoftwareStatus Status;
      attribute string Name;
      void StartUp ( )
         raises (AMS_Util::AMS_Error);
      void ShutDown ( )
         raises (AMS_Util::AMS_Error);
   };

   //
   interface AMS_LoadBalancingGroup : AMS_CIM::CIM_RedundancyGroup {
      readonly attribute AMS_Application Owner;
      readonly attribute AMS_ApplicationSpecification::AMS_SoftwareFeatureSpec
                                                    LoadBalancingFeature;
      AMS_ExecutableSoftwareElementListIterator GetRedundancyComponentIterator ( );
      readonly attribute AMS_ExecutableSoftwareElementList RedundancyComponent;
      attribute AMS_BalancingStyle Style;
      attribute AMS_RTSoftwareStatus Status;
      attribute string Name;
      void StartUp ( )
         raises (AMS_Util::AMS_Error);
      void ShutDown ( )
         raises (AMS_Util::AMS_Error);
   };

};

#endif/* _AMS_Application_IDL_ */
```

8.6.5 AMS_ApplicationDeployment.idl

```
// Copyright 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
// Themis Computer and Progeny Systems Corporation.

#include "AMS_Util.idl"
#include "AMS_CIM.idl"
#ifndef _AMS_ApplicationDeployment_IDL_
#define _AMS_ApplicationDeployment_IDL_

#pragma prefix "omg.org"

#include "AMS_ApplicationSpecification.idl"
#include "AMS_ApplicationDeploymentSpecification.idl"
#include "AMS_LogicalHardware.idl"

module AMS_ApplicationModule {
   interface AMS_ExecutableSoftwareElement;
};

module AMS_ApplicationDeployment {
   //
   const string AMS_DeploymentLink_CLASSID
         = "IDL:omg.org/AMS_ApplicationDeployment/AMS_DeploymentLink:1.0";
   interface AMS_DeploymentLink ;
   typedef sequence<AMS_DeploymentLink> AMS_DeploymentLinkList;
   interface AMS_DeploymentLinkListIterator : AMS_Util::AMS_Iterator {
      AMS_DeploymentLink GetCurrentItem ()
         raises (AMS_Util::AMS_NoSuchElementException);
   };
```

```
   //
   const string AMS_DeploymentConfiguration_CLASSID
         = "IDL:omg.org/AMS_ApplicationDeployment/AMS_DeploymentConfiguration:1.0";
   interface AMS_DeploymentConfiguration ;
   typedef sequence<AMS_DeploymentConfiguration> AMS_DeploymentConfigurationList;
   interface AMS_DeploymentConfigurationListIterator : AMS_Util::AMS_Iterator {
      AMS_DeploymentConfiguration GetCurrentItem ()
         raises (AMS_Util::AMS_NoSuchElementException);
   };

   //
   interface AMS_DeploymentLink : AMS_CIM::CIM_LogicalElement {
      readonly attribute AMS_LogicalHardware::AMS_Host HostUsed;
      readonly attribute AMS_ApplicationModule::AMS_ExecutableSoftwareElement
ESEDeployed;
      readonly attribute AMS_DeploymentConfiguration Owner;
      attribute string LinkID;
   };

   //
   interface AMS_DeploymentConfiguration : AMS_CIM::CIM_LogicalElement {
      AMS_DeploymentLinkListIterator GetDeploymentLinksIterator ( );
      readonly attribute AMS_DeploymentLinkList DeploymentLinks;
      readonly attribute AMS_ApplicationDeploymentSpecification::AMS_DeploymentSpec
                                                  DeploymentSpecAssoc;
      attribute string Name;
      void StartUp ( )
         raises (AMS_Util::AMS_Error);
      void ShutDown ( )
         raises (AMS_Util::AMS_Error);
   };

};

#endif/* _AMS_ApplicationDeployment_IDL_ */
```

8.6.6 AMS_ApplicationDeploymentSpecification.idl

```
// Copyright 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
// Themis Computer and Progeny Systems Corporation.

#include "AMS_Util.idl"
#include "AMS_CIM.idl"

#ifndef _AMS_ApplicationDeploymentSpecification_IDL_
#define _AMS_ApplicationDeploymentSpecification_IDL_

#pragma prefix "omg.org"

#include "AMS_LogicalHardware.idl"
#include "AMS_LogicalHardwareSpecification.idl"
#include "AMS_ApplicationSpecification.idl"

module AMS_ApplicationDeployment {
   interface AMS_DeploymentConfiguration;
   typedef sequence<AMS_DeploymentConfiguration> AMS_DeploymentConfigurationList;
   interface AMS_DeploymentConfigurationListIterator;
};

module AMS_ApplicationDeploymentSpecification {
```

```
      //
      const string AMS_DeploymentLinkSpec_CLASSID
        = "IDL:omg.org/AMS_ApplicationDeploymentSpecification/AMS_DeploymentLinkSpec:1.0";
      interface AMS_DeploymentLinkSpec ;
      typedef sequence<AMS_DeploymentLinkSpec> AMS_DeploymentLinkSpecList;
      interface AMS_DeploymentLinkSpecListIterator : AMS_Util::AMS_Iterator {
         AMS_DeploymentLinkSpec GetCurrentItem ()
            raises (AMS_Util::AMS_NoSuchElementException);
      };

      //
      const string AMS_DeploymentSpec_CLASSID
         = "IDL:omg.org/AMS_ApplicationDeploymentSpecification/AMS_DeploymentSpec:1.0";
      interface AMS_DeploymentSpec ;
      typedef sequence<AMS_DeploymentSpec> AMS_DeploymentSpecList;
      interface AMS_DeploymentSpecListIterator : AMS_Util::AMS_Iterator {
         AMS_DeploymentSpec GetCurrentItem ()
            raises (AMS_Util::AMS_NoSuchElementException);
      };

      //
      struct AMS_ActionOnLink {
         AMS_ApplicationSpecification::AMS_ActionCheckCase _Case;
      };

      //
      interface AMS_DeploymentLinkSpec : AMS_CIM::CIM_LogicalElement {
         readonly attribute
            AMS_LogicalHardwareSpecification::AMS_ConfigurationSpecification
                                                                  ConfSpecDLS;
         readonly attribute AMS_ApplicationSpecification::AMS_ESESpec SEDeployed;
         AMS_LogicalHardware::AMS_HostListIterator GetHostUsedIterator ( );
         readonly attribute AMS_LogicalHardware::AMS_HostList HostUsed;
         readonly attribute AMS_DeploymentLinkSpec DeploymentLinkDependency;
         readonly attribute AMS_DeploymentSpec Owner;
         // association class
         readonly attribute AMS_CIM::CIM_Action ActionOnLink;
         AMS_CIM::CIM_Action GetActionOnLink (
                        in AMS_ApplicationSpecification::AMS_ActionCheckCase state );
         //
         AMS_LogicalHardware::AMS_OperatingSystemListIterator GetOSUsedIterator ( );
         readonly attribute AMS_LogicalHardware::AMS_OperatingSystemList OSUsed;
         attribute string LinkID;
      };

      //
      interface AMS_DeploymentSpec : AMS_CIM::CIM_LogicalElement {
         AMS_ApplicationDeployment::AMS_DeploymentConfigurationListIterator
                                          GetDeploymentSpecAssocIterator ( );
         readonly attribute AMS_ApplicationDeployment::AMS_DeploymentConfigurationList
                                          DeploymentSpecAssoc;
         AMS_DeploymentLinkSpecListIterator GetDeploymentSpecLinksIterator ( );
         readonly attribute AMS_DeploymentLinkSpecList DeploymentSpecLinks;
         attribute string Name;
         void Deploy ( )
            raises (AMS_Util::AMS_Error);
      };

   };
```

```
#endif/* _AMS_ApplicationDeploymentSpecification_IDL_ */
```

8.6.7 AMS_ApplicationSpecification.idl

```
// Copyright 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
// Themis Computer and Progeny Systems Corporation.

#include "AMS_Util.idl"
#include "AMS_CIM.idl"

#ifndef _AMS_ApplicationSpecification_IDL_
#define _AMS_ApplicationSpecification_IDL_

#pragma prefix "omg.org"

#include "AMS_SupportedApplicationModel.idl"

module AMS_ApplicationSpecification {
   //
   enum AMS_TypeOfFeature {
      SYSTEM,
      APPLICATION,
      REDUNDANCY_GROUP,
      LOADBALANCING_GROUP
   };
   typedef sequence<AMS_TypeOfFeature> AMS_TypeOfFeatureList;
   //
   enum AMS_ActionCheckCase {
      CASE_PRE_DEPLOY,
      CASE_DEPLOY,
      CASE_POST_DEPLOY,
      CASE_PRE_START,
      CASE_START,
      CASE_POST_START,
      CASE_PRE_SHUTDOWN,
      CASE_SHUTDOWN,
      CASE_POST_SHUTDOWN,
      CASE_ALTERNATE_SHUTDOWN
   };
   typedef sequence<AMS_ActionCheckCase> AMS_ActionCheckCaseList;
   //
   enum AMS_RedundancyEltState {
      REDSTATE_NORG,
      REDSTATE_PRIMARY,
      REDSTATE_PASSIVE
   };
   typedef sequence<AMS_RedundancyEltState> AMS_RedundancyEltStateList;
   //
   const string AMS_SoftwareFeatureSpec_CLASSID
         = "IDL:omg.org/AMS_ApplicationSpecification/AMS_SoftwareFeatureSpec:1.0";
   interface AMS_SoftwareFeatureSpec ;
   typedef sequence<AMS_SoftwareFeatureSpec> AMS_SoftwareFeatureSpecList;
   interface AMS_SoftwareFeatureSpecListIterator : AMS_Util::AMS_Iterator {
      AMS_SoftwareFeatureSpec GetCurrentItem ()
         raises (AMS_Util::AMS_NoSuchElementException);
   };

   //
   const string AMS_ESESpec_CLASSID
         = "IDL:omg.org/AMS_ApplicationSpecification/AMS_ESESpec:1.0";
```

```
interface AMS_ESESpec ;
typedef sequence<AMS_ESESpec> AMS_ESESpecList;
interface AMS_ESESpecListIterator : AMS_Util::AMS_Iterator {
   AMS_ESESpec GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_ExecuteProgram_CLASSID
      = "IDL:omg.org/AMS_ApplicationSpecification/AMS_ExecuteProgram:1.0";
interface AMS_ExecuteProgram ;
typedef sequence<AMS_ExecuteProgram> AMS_ExecuteProgramList;
interface AMS_ExecuteProgramListIterator : AMS_Util::AMS_Iterator {
   AMS_ExecuteProgram GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_MechanizedAction_CLASSID
      = "IDL:omg.org/AMS_ApplicationSpecification/AMS_MechanizedAction:1.0";
interface AMS_MechanizedAction ;
typedef sequence<AMS_MechanizedAction> AMS_MechanizedActionList;
interface AMS_MechanizedActionListIterator : AMS_Util::AMS_Iterator {
   AMS_MechanizedAction GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_Property_StdMechanism_CLASSID
      = "IDL:omg.org/AMS_ApplicationSpecification/AMS_Property_StdMechanism:1.0";
interface AMS_Property_StdMechanism;
typedef sequence<AMS_Property_StdMechanism> AMS_Property_StdMechanismList;
interface AMS_Property_StdMechanismListIterator : AMS_Util::AMS_Iterator {
   AMS_Property_StdMechanism GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
struct AMS_SEShutdownDependency {
   AMS_Util::AMS_uint16 TimeSinceShutdown;
};

//
struct AMS_SEStartTimeDependency {
   AMS_Util::AMS_uint16 TimeSinceStartup;
};

//
struct AMS_SEStartCPUDependency {
   AMS_Util::AMS_uint16 CPUload;
};

//
const string AMS_CCMDeploy_CLASSID
      = "IDL:omg.org/AMS_ApplicationSpecification/AMS_CCMDeploy:1.0";
interface AMS_CCMDeploy ;
typedef sequence<AMS_CCMDeploy> AMS_CCMDeployList;
interface AMS_CCMDeployListIterator: AMS_Util::AMS_Iterator {
   AMS_CCMDeploy GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
```

```
};

//
const string AMS_CCMStart_CLASSID
      = "IDL:omg.org/AMS_ApplicationSpecification/AMS_CCMStart:1.0";
interface AMS_CCMStart ;
typedef sequence<AMS_CCMStart> AMS_CCMStartList;
interface AMS_CCMStartListIterator: AMS_Util::AMS_Iterator {
   AMS_CCMStart GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_CCMStop_CLASSID
      = "IDL:omg.org/AMS_ApplicationSpecification/AMS_CCMStop:1.0";
interface AMS_CCMStop;
typedef sequence<AMS_CCMStop> AMS_CCMStopList;
interface AMS_CCMStopListIterator: AMS_Util::AMS_Iterator {
   AMS_CCMStop GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_J2EEDeploy_CLASSID
      = "IDL:omg.org/AMS_ApplicationSpecification/AMS_J2EEDeploy:1.0";
interface AMS_J2EEDeploy;
typedef sequence<AMS_J2EEDeploy> AMS_J2EEDeployList;
interface AMS_J2EEDeployListIterator: AMS_Util::AMS_Iterator {
   AMS_J2EEDeploy GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_J2EEStart_CLASSID
      = "IDL:omg.org/AMS_ApplicationSpecification/AMS_J2EEStart:1.0";
interface AMS_J2EEStart ;
typedef sequence<AMS_J2EEStart> AMS_J2EEStartList;
interface AMS_J2EEStartListIterator: AMS_Util::AMS_Iterator {
   AMS_J2EEStart GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_J2EEStop_CLASSID
      = "IDL:omg.org/AMS_ApplicationSpecification/AMS_J2EEStop:1.0";
interface AMS_J2EEStop;
typedef sequence<AMS_J2EEStop> AMS_J2EEStopList;
interface AMS_J2EEStopListIterator: AMS_Util::AMS_Iterator {
   AMS_J2EEStop GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_SecurityCheck CLASSID
      = "IDL:omg.org/AMS_ApplicationSpecification/AMS_SecurityCheck:1.0";
interface AMS_SecurityCheck;
typedef sequence<AMS_SecurityCheck > AMS_SecurityCheck List;
interface AMS_ SecurityCheck ListIterator: AMS_Util::AMS_Iterator {
   AMS_SecurityCheck GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
```

```
};

//
interface AMS_SoftwareFeatureSpec : AMS_CIM::CIM_SoftwareFeature {
   readonly attribute AMS_SoftwareFeatureSpec Owner;
   AMS_SoftwareFeatureSpecListIterator GetFeatureOfFeatureIterator ( );
   readonly attribute AMS_SoftwareFeatureSpecList FeatureOfFeature;
   AMS_ESESpecListIterator GetSoftwareFeatureSoftwareElementIterator ( );
   readonly attribute AMS_ESESpecList SoftwareFeatureSoftwareElement;
   attribute AMS_TypeOfFeature TypeOfFeature;
   // association class
   AMS_CIM::CIM_CheckListIterator GetSoftwareFeatureCheckIterator ( );
   readonly attribute AMS_CIM::CIM_CheckList SoftwareFeatureCheck;
   AMS_CIM::CIM_CheckList GetSoftwareFeatureCheckOnCase (
                                    in AMS_ActionCheckCase state);
   AMS_CIM::CIM_CheckListIterator GetSoftwareFeatureCheckIteratorOnCase (
                                    in AMS_ActionCheckCase state);
   // association class
   AMS_CIM::CIM_ActionListIterator GetSoftwareFeatureActionIterator ( );
   readonly attribute AMS_CIM::CIM_ActionList SoftwareFeatureAction;
   AMS_CIM::CIM_ActionList GetSoftwareFeatureAction (
                                    in AMS_ActionCheckCase state);
   AMS_CIM::CIM_ActionListIterator GetSoftwareFeatureActionIteratorOnCase (
                                    in AMS_ActionCheckCase state);
   //
   attribute string Name;
};

//
interface AMS_ESESpec : AMS_CIM::CIM_SoftwareElement {
   readonly attribute AMS_SoftwareFeatureSpec Owner;
   attribute AMS_SAM::AMS_ModelType ModelType;
   // association class
   AMS_ESESpecListIterator GetSEStartDependencyIterator ( );
   readonly attribute AMS_ESESpecList SEStartDependency;
   AMS_ESESpecList GetSEStartCPUDependencyOnCase (
                                    in AMS_SEStartCPUDependency state );
   AMS_ESESpecListIterator GetSEStartCPUDependencyIteratorOnCase (
                                    in AMS_SEStartCPUDependency state );
   AMS_ESESpecList GetSEStartTimeDependencyOnCase (
                                    in AMS_SEStartTimeDependency state );
   AMS_ESESpecListIterator GetSEStartTimeDependencyIteratorOnCase (
                                    in AMS_SEStartTimeDependency state );
   // association class
   AMS_CIM::CIM_CheckListIterator GetSoftwareElementCheckIterator ( );
   readonly attribute AMS_CIM::CIM_CheckList SoftwareElementCheck;
   AMS_CIM::CIM_CheckList GetSoftwareElementCheckOnCase (
                                      in AMS_ActionCheckCase state );
   AMS_CIM::CIM_CheckListIterator GetSoftwareElementCheckIteratorOnCase (
                                      in AMS_ActionCheckCase state );
   // association class
   AMS_CIM::CIM_ActionListIterator GetSoftwareElementActionIterator ( );
   readonly attribute AMS_CIM::CIM_ActionList SoftwareElementAction;
   AMS_CIM::CIM_ActionList GetSoftwareElementActionOnCase (
                                      in AMS_ActionCheckCase state);
   AMS_CIM::CIM_ActionListIterator GetSoftwareElementActionIteratorOnCase (
                                      in AMS_ActionCheckCase state);
   //
   attribute AMS_RedundancyEltState RedInitState;
   // association class
```

```
   AMS_ESESpecListIterator GetSEShutdownDependencyIterator ( );
   readonly attribute AMS_ESESpecList SEShutdownDependency;
   AMS_ESESpecList GetSEShutdownDependencyOnCase (
                                        in AMS_SEShutdownDependency state );
   AMS_ESESpecListIterator GetSEShutdownDependencyIteratorOnCase (
                                        in AMS_SEShutdownDependency state );
   //
   attribute string Name;
   attribute string SoftwareElementID;
};

//
interface AMS_ApplicationModelCheck : AMS_CIM::CIM_Check {
   readonly attribute AMS_SAM::AMS_SupportedApplicationModel SAMCheck;
};

//
interface AMS_SecurityCheck : AMS_CIM::CIM_Check {
};

//
interface AMS_ExecuteProgram : AMS_CIM::CIM_ExecuteProgram {
   attribute string Environment;
};

//
interface AMS_CCMDeploy: AMS_CIM::CIM_Action {
   attribute string ComponentPackageDescriptor;
};

//
interface AMS_CCMStart: AMS_CIM::CIM_Action {
};

//
interface AMS_CCMStop: AMS_CIM::CIM_Action {
};

//
interface AMS_J2EEDeploy: AMS_CIM::CIM_Action {
   attribute string EnterpriseArchive;
};

//
interface AMS_J2EEStart: AMS_CIM::CIM_Action {
};

//
interface AMS_J2EEStop: AMS_CIM::CIM_Action {
};

//
interface AMS_Property_StdMechanism {
   attribute string Name;
   attribute string Value;
   attribute string InstanceID;
   attribute AMS_SAM::AMS_StdMechanism StdName;
};

//
```

```
   interface AMS_MechanizedAction: AMS_CIM::CIM_Action {
      attribute AMS_Property_StdMechanism ActionMechanism;
   };
};

#endif/* _AMS_ApplicationSpecification_IDL_ */
```

8.6.8 AMS_CIM.idl

```
// Copyright 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
// Themis Computer and Progeny Systems Corporation.

#include "AMS_Util.idl"

#ifndef _AMS_CIM_IDL_
#define _AMS_CIM_IDL_

#pragma prefix "omg.org"

module AMS_LogicalHardware {
   interface AMS_ComputerSystem;
   typedef sequence<AMS_ComputerSystem> AMS_ComputerSystemList;
   interface AMS_ComputerSystemListIterator;
} ;

module AMS_ApplicationSpecification {
   //
   enum AMS_ActionCheckCase {
      CASE_PRE_DEPLOY,
      CASE_DEPLOY,
      CASE_POST_DEPLOY,
      CASE_PRE_START,
      CASE_START,
      CASE_POST_START,
      CASE_PRE_SHUTDOWN,
      CASE_SHUTDOWN,
      CASE_POST_SHUTDOWN,
      CASE_ALTERNATE_SHUTDOWN
   };
   typedef sequence<AMS_ActionCheckCase> AMS_ActionCheckCaseList;
};

module AMS_CIM {
   //
   const string CIM_System_CLASSID
         = "IDL:omg.org/AMS_CIM/CIM_System:1.0";
   interface CIM_System ;
   typedef sequence<CIM_System> CIM_SystemList;
   interface CIM_SystemListIterator : AMS_Util::AMS_Iterator {
      CIM_System GetCurrentItem ()
         raises (AMS_Util::AMS_NoSuchElementException);
   };

   //
   const string CIM_ComputerSystem_CLASSID
         = "IDL:omg.org/AMS_CIM/CIM_ComputerSystem:1.0";
   interface CIM_ComputerSystem ;
   typedef sequence<CIM_ComputerSystem> CIM_ComputerSystemList;
   interface CIM_ComputerSystemListIterator : AMS_Util::AMS_Iterator {
      CIM_ComputerSystem GetCurrentItem ()
```

```
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_IPProtocolEndPoint_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_IPProtocolEndPoint:1.0";
interface CIM_IPProtocolEndPoint ;
typedef sequence<CIM_IPProtocolEndPoint> CIM_IPProtocolEndPointList;
interface CIM_IPProtocolEndPointListIterator : AMS_Util::AMS_Iterator {
   CIM_IPProtocolEndPoint GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_NextHopRoute_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_NextHopRoute:1.0";
interface CIM_NextHopRoute ;
typedef sequence<CIM_NextHopRoute> CIM_NextHopRouteList;
interface CIM_NextHopRouteListIterator : AMS_Util::AMS_Iterator {
   CIM_NextHopRoute GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_LogicalDevice_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_LogicalDevice:1.0";
interface CIM_LogicalDevice ;
typedef sequence<CIM_LogicalDevice> CIM_LogicalDeviceList;
interface CIM_LogicalDeviceListIterator : AMS_Util::AMS_Iterator {
   CIM_LogicalDevice GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Location_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Location:1.0";
interface CIM_Location ;
typedef sequence<CIM_Location> CIM_LocationList;
interface CIM_LocationListIterator : AMS_Util::AMS_Iterator {
   CIM_Location GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_LogicalElement_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_LogicalElement:1.0";
interface CIM_LogicalElement ;
typedef sequence<CIM_LogicalElement> CIM_LogicalElementList;
interface CIM_LogicalElementListIterator : AMS_Util::AMS_Iterator {
   CIM_LogicalElement GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_OperatingSystem_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_OperatingSystem:1.0";
interface CIM_OperatingSystem ;
typedef sequence<CIM_OperatingSystem> CIM_OperatingSystemList;
interface CIM_OperatingSystemListIterator : AMS_Util::AMS_Iterator {
   CIM_OperatingSystem GetCurrentItem ()
```

```
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_EnabledLogicalElement_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_EnabledLogicalElement:1.0";
interface CIM_EnabledLogicalElement ;
typedef sequence<CIM_EnabledLogicalElement> CIM_EnabledLogicalElementList;
interface CIM_EnabledLogicalElementListIterator : AMS_Util::AMS_Iterator {
   CIM_EnabledLogicalElement GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_ApplicationSystem_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_ApplicationSystem:1.0";
interface CIM_ApplicationSystem ;
typedef sequence<CIM_ApplicationSystem> CIM_ApplicationSystemList;
interface CIM_ApplicationSystemListIterator : AMS_Util::AMS_Iterator {
   CIM_ApplicationSystem GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Service_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Service:1.0";
interface CIM_Service ;
typedef sequence<CIM_Service> CIM_ServiceList;
interface CIM_ServiceListIterator : AMS_Util::AMS_Iterator {
   CIM_Service GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_SoftwareFeature_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_SoftwareFeature:1.0";
interface CIM_SoftwareFeature ;
typedef sequence<CIM_SoftwareFeature> CIM_SoftwareFeatureList;
interface CIM_SoftwareFeatureListIterator : AMS_Util::AMS_Iterator {
   CIM_SoftwareFeature GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_SoftwareElement_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_SoftwareElement:1.0";
interface CIM_SoftwareElement ;
typedef sequence<CIM_SoftwareElement> CIM_SoftwareElementList;
interface CIM_SoftwareElementListIterator : AMS_Util::AMS_Iterator {
   CIM_SoftwareElement GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_OSVersionCheck_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_OSVersionCheck:1.0";
interface CIM_OSVersionCheck ;
typedef sequence<CIM_OSVersionCheck> CIM_OSVersionCheckList;
interface CIM_OSVersionCheckListIterator : AMS_Util::AMS_Iterator {
   CIM_OSVersionCheck GetCurrentItem ()
```

```
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Check_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Check:1.0";
interface CIM_Check ;
typedef sequence<CIM_Check> CIM_CheckList;
interface CIM_CheckListIterator : AMS_Util::AMS_Iterator {
   CIM_Check GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_ArchitectureCheck_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_ArchitectureCheck:1.0";
interface CIM_ArchitectureCheck ;
typedef sequence<CIM_ArchitectureCheck> CIM_ArchitectureCheckList;
interface CIM_ArchitectureCheckListIterator : AMS_Util::AMS_Iterator {
   CIM_ArchitectureCheck GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Action_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Action:1.0";
interface CIM_Action ;
typedef sequence<CIM_Action> CIM_ActionList;
interface CIM_ActionListIterator : AMS_Util::AMS_Iterator {
   CIM_Action GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_ExecuteProgram_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_ExecuteProgram:1.0";
interface CIM_ExecuteProgram ;
typedef sequence<CIM_ExecuteProgram> CIM_ExecuteProgramList;
interface CIM_ExecuteProgramListIterator : AMS_Util::AMS_Iterator {
   CIM_ExecuteProgram GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_FileAction_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_FileAction:1.0";
interface CIM_FileAction ;
typedef sequence<CIM_FileAction> CIM_FileActionList;
interface CIM_FileActionListIterator : AMS_Util::AMS_Iterator {
   CIM_FileAction GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_CopyFileAction_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_CopyFileAction:1.0";
interface CIM_CopyFileAction ;
typedef sequence<CIM_CopyFileAction> CIM_CopyFileActionList;
interface CIM_CopyFileActionListIterator : AMS_Util::AMS_Iterator {
   CIM_CopyFileAction GetCurrentItem ()
```

```
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_DirectorySpecification_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_DirectorySpecification:1.0";
interface CIM_DirectorySpecification ;
typedef sequence<CIM_DirectorySpecification> CIM_DirectorySpecificationList;
interface CIM_DirectorySpecificationListIterator : AMS_Util::AMS_Iterator {
   CIM_DirectorySpecification GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_RedundancyGroup_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_RedundancyGroup:1.0";
interface CIM_RedundancyGroup ;
typedef sequence<CIM_RedundancyGroup> CIM_RedundancyGroupList;
interface CIM_RedundancyGroupListIterator : AMS_Util::AMS_Iterator {
   CIM_RedundancyGroup GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_ProtocolEndPoint_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_ProtocolEndPoint:1.0";
interface CIM_ProtocolEndPoint ;
typedef sequence<CIM_ProtocolEndPoint> CIM_ProtocolEndPointList;
interface CIM_ProtocolEndPointListIterator : AMS_Util::AMS_Iterator {
   CIM_ProtocolEndPoint GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_AdminDomain_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_AdminDomain:1.0";
interface CIM_AdminDomain ;
typedef sequence<CIM_AdminDomain> CIM_AdminDomainList;
interface CIM_AdminDomainListIterator : AMS_Util::AMS_Iterator {
   CIM_AdminDomain GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_LogicalDisk_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_LogicalDisk:1.0";
interface CIM_LogicalDisk ;
typedef sequence<CIM_LogicalDisk> CIM_LogicalDiskList;
interface CIM_LogicalDiskListIterator : AMS_Util::AMS_Iterator {
   CIM_LogicalDisk GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Memory_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Memory:1.0";
interface CIM_Memory ;
typedef sequence<CIM_Memory> CIM_MemoryList;
interface CIM_MemoryListIterator : AMS_Util::AMS_Iterator {
   CIM_Memory GetCurrentItem ()
```

```
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_PowerSupply_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_PowerSupply:1.0";
interface CIM_PowerSupply ;
typedef sequence<CIM_PowerSupply> CIM_PowerSupplyList;
interface CIM_PowerSupplyListIterator : AMS_Util::AMS_Iterator {
   CIM_PowerSupply GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Watchdog_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Watchdog:1.0";
interface CIM_Watchdog ;
typedef sequence<CIM_Watchdog> CIM_WatchdogList;
interface CIM_WatchdogListIterator : AMS_Util::AMS_Iterator {
   CIM_Watchdog GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Processor_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Processor:1.0";
interface CIM_Processor ;
typedef sequence<CIM_Processor> CIM_ProcessorList;
interface CIM_ProcessorListIterator : AMS_Util::AMS_Iterator {
   CIM_Processor GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_LogicalPort_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_LogicalPort:1.0";
interface CIM_LogicalPort ;
typedef sequence<CIM_LogicalPort> CIM_LogicalPortList;
interface CIM_LogicalPortListIterator : AMS_Util::AMS_Iterator {
   CIM_LogicalPort GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_NetworkPort_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_NetworkPort:1.0";
interface CIM_NetworkPort ;
typedef sequence<CIM_NetworkPort> CIM_NetworkPortList;
interface CIM_NetworkPortListIterator : AMS_Util::AMS_Iterator {
   CIM_NetworkPort GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_EthernetPort_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_EthernetPort:1.0";
interface CIM_EthernetPort ;
typedef sequence<CIM_EthernetPort> CIM_EthernetPortList;
interface CIM_EthernetPortListIterator : AMS_Util::AMS_Iterator {
   CIM_EthernetPort GetCurrentItem ()
```

```
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Display_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Display:1.0";
interface CIM_Display ;
typedef sequence<CIM_Display> CIM_DisplayList;
interface CIM_DisplayListIterator : AMS_Util::AMS_Iterator {
   CIM_Display GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Sensor_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Sensor:1.0";
interface CIM_Sensor ;
typedef sequence<CIM_Sensor> CIM_SensorList;
interface CIM_SensorListIterator : AMS_Util::AMS_Iterator {
   CIM_Sensor GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_StorageExtent_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_StorageExtent:1.0";
interface CIM_StorageExtent ;
typedef sequence<CIM_StorageExtent> CIM_StorageExtentList;
interface CIM_StorageExtentListIterator : AMS_Util::AMS_Iterator {
   CIM_StorageExtent GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_PhysicalElement_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_PhysicalElement:1.0";
interface CIM_PhysicalElement ;
typedef sequence<CIM_PhysicalElement> CIM_PhysicalElementList;
interface CIM_PhysicalElementListIterator : AMS_Util::AMS_Iterator {
   CIM_PhysicalElement GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Log_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Log:1.0";
interface CIM_Log ;
typedef sequence<CIM_Log> CIM_LogList;
interface CIM_LogListIterator : AMS_Util::AMS_Iterator {
   CIM_Log GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_ManagedElement_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_ManagedElement:1.0";
interface CIM_ManagedElement ;
typedef sequence<CIM_ManagedElement> CIM_ManagedElementList;
interface CIM_ManagedElementListIterator : AMS_Util::AMS_Iterator {
   CIM_ManagedElement GetCurrentItem ()
```

```
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_RecordForLog_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_RecordForLog:1.0";
interface CIM_RecordForLog ;
typedef sequence<CIM_RecordForLog> CIM_RecordForLogList;
interface CIM_RecordForLogListIterator : AMS_Util::AMS_Iterator {
   CIM_RecordForLog GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Dependency_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Dependency:1.0";
interface CIM_Dependency ;
typedef sequence<CIM_Dependency> CIM_DependencyList;
interface CIM_DependencyListIterator : AMS_Util::AMS_Iterator {
   CIM_Dependency GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_ConnectivityCollection_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_ConnectivityCollection:1.0";
interface CIM_ConnectivityCollection ;
typedef sequence<CIM_ConnectivityCollection> CIM_ConnectivityCollectionList;
interface CIM_ConnectivityCollectionListIterator : AMS_Util::AMS_Iterator {
   CIM_ConnectivityCollection GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_EthernetPortStatistics_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_EthernetPortStatistics:1.0";
interface CIM_EthernetPortStatistics ;
typedef sequence<CIM_EthernetPortStatistics> CIM_EthernetPortStatisticsList;
interface CIM_EthernetPortStatisticsListIterator : AMS_Util::AMS_Iterator {
   CIM_EthernetPortStatistics GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_LANEndPoint_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_LANEndPoint:1.0";
interface CIM_LANEndPoint ;
typedef sequence<CIM_LANEndPoint> CIM_LANEndPointList;
interface CIM_LANEndPointListIterator : AMS_Util::AMS_Iterator {
   CIM_LANEndPoint GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Network_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Network:1.0";
interface CIM_Network ;
typedef sequence<CIM_Network> CIM_NetworkList;
interface CIM_NetworkListIterator : AMS_Util::AMS_Iterator {
   CIM_Network GetCurrentItem ()
```

```
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_NextHopIPRoute_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_NextHopIPRoute:1.0";
interface CIM_NextHopIPRoute ;
typedef sequence<CIM_NextHopIPRoute> CIM_NextHopIPRouteList;
interface CIM_NextHopIPRouteListIterator : AMS_Util::AMS_Iterator {
   CIM_NextHopIPRoute GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_RemoteServiceAccessPoint_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_RemoteServiceAccessPoint:1.0";
interface CIM_RemoteServiceAccessPoint ;
typedef sequence<CIM_RemoteServiceAccessPoint> CIM_RemoteServiceAccessPointList;
interface CIM_RemoteServiceAccessPointListIterator : AMS_Util::AMS_Iterator {
   CIM_RemoteServiceAccessPoint GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_ServiceAccessPoint_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_ServiceAccessPoint:1.0";
interface CIM_ServiceAccessPoint ;
typedef sequence<CIM_ServiceAccessPoint> CIM_ServiceAccessPointList;
interface CIM_ServiceAccessPointListIterator : AMS_Util::AMS_Iterator {
   CIM_ServiceAccessPoint GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Process_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Process:1.0";
interface CIM_Process ;
typedef sequence<CIM_Process> CIM_ProcessList;
interface CIM_ProcessListIterator : AMS_Util::AMS_Iterator {
   CIM_Process GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string CIM_Thread_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_Thread:1.0";
interface CIM_Thread ;
typedef sequence<CIM_Thread> CIM_ThreadList;
interface CIM_ThreadListIterator : AMS_Util::AMS_Iterator {
   CIM_Thread GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};


//
const string CIM_UnixProcess_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_UnixProcess:1.0";
interface CIM_UnixProcess ;
typedef sequence<CIM_UnixProcess> CIM_UnixProcessList;
interface CIM_UnixProcessListIterator : AMS_Util::AMS_Iterator {
```

```
   CIM_UnixProcess GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};


//
const string CIM_UnixThread_CLASSID
      = "IDL:omg.org/AMS_CIM/CIM_UnixThread:1.0";
interface CIM_UnixThread ;
typedef sequence<CIM_UnixThread> CIM_UnixThreadList;
interface CIM_UnixThreadListIterator : AMS_Util::AMS_Iterator {
   CIM_UnixThread GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
interface CIM_LogicalElement {
};

//
interface CIM_EnabledLogicalElement : CIM_LogicalElement {
};

//
interface CIM_System : CIM_EnabledLogicalElement {
};

//
interface CIM_ComputerSystem : CIM_System {
};

//
interface CIM_ServiceAccessPoint {
   // AMS_LogicalHardware::AMS_ComputerSystem or CIM_ConnectivityCollection
   readonly attribute Object Owner;
   //
   CIM_LANEndPointListIterator GetBindsToLANEndPointIterator ( );
   readonly attribute CIM_LANEndPointList BindsToLANEndPoint;
   attribute string Name;
};

//
interface CIM_RemoteServiceAccessPoint : CIM_ServiceAccessPoint {
   CIM_NextHopRouteListIterator GetAssociatedNextHopIterator ( );
   readonly attribute CIM_NextHopRouteList AssociatedNextHop;
   attribute string AccessInfo;
   attribute AMS_Util::AMS_uint16 InfoFormat;
};

//
interface CIM_ProtocolEndPoint : CIM_ServiceAccessPoint {
   CIM_ProtocolEndPointListIterator GetEndpointIdentityIterator ( );
   readonly attribute CIM_ProtocolEndPointList EndpointIdentity;
   CIM_NextHopRouteListIterator GetRouteUsesEndpointIterator ( );
   readonly attribute CIM_NextHopRouteList RouteUsesEndpoint;
};

//
interface CIM_LANEndPoint : CIM_ProtocolEndPoint {
   CIM_EthernetPortStatisticsListIterator GetElementStatisticalDataIterator ( );
```

```
   readonly attribute CIM_EthernetPortStatisticsList ElementStatisticalData;
   CIM_ServiceAccessPointListIterator GetBindsToLANEndPointIteratorRev ( );
   readonly attribute CIM_ServiceAccessPointList BindsToLANEndPointRev;
   attribute string LANID;
   attribute string MACAddress;
   attribute string AliasAddresses;
   attribute string GroupAddresses;
   attribute AMS_Util::AMS_uint32 MaxDataSize;
};

//
interface CIM_IPProtocolEndPoint : CIM_ProtocolEndPoint {
   attribute string IPv4Address;
   attribute string IPv6Address;
   attribute string SubnetMask;
   attribute AMS_Util::AMS_uint8 PrefixLength;
};

//
interface CIM_NextHopRoute {
   readonly attribute CIM_RemoteServiceAccessPoint AssociatedNextHop;
   readonly attribute CIM_ProtocolEndPoint RouteUsesEndpoint;
   // AMS_Host or AMS_Router
   readonly attribute Object Owner;
   //
   attribute string InstanceID;
   attribute string DestinationAddress;
   attribute AMS_Util::AMS_uint16 AdminDistance;
   attribute AMS_Util::AMS_uint16 RouteMetric;
   attribute boolean IsStatic;
   attribute AMS_Util::AMS_uint16 TypeOfRoute;
};

//
interface CIM_LogicalDevice {
   CIM_PhysicalElementListIterator GetRealizesIterator ( );
   readonly attribute CIM_PhysicalElementList Realizes;
   readonly attribute AMS_LogicalHardware::AMS_ComputerSystem Owner;
   attribute string DeviceID;
};

//
interface CIM_Location {
   CIM_PhysicalElementListIterator GetPhysicalElementLocationIterator ( );
   readonly attribute CIM_PhysicalElementList PhysicalElementLocation;
   AMS_LogicalHardware::AMS_ComputerSystemListIterator
                                       GetElementLocationIterator ( );
   readonly attribute AMS_LogicalHardware::AMS_ComputerSystemList ElementLocation;
   attribute string Name;
};

//
interface CIM_OperatingSystem {
};

//
interface CIM_ApplicationSystem : CIM_System {
};

//
```

```
interface CIM_Service : CIM_EnabledLogicalElement {
};

//
interface CIM_SoftwareFeature : CIM_LogicalElement {
};

//
interface CIM_SoftwareElement : CIM_LogicalElement {
};

//
interface CIM_Check {
   // AMS_ApplicationSpecification::AMS_ESESpec
   // or AMS_ApplicationSpecification::AMS_SoftwareFeatureSpec
   // or CIM_CopyFileAction
   readonly attribute Object Owner;
   readonly attribute AMS_ApplicationSpecification::AMS_ActionCheckCase state;
   //
   attribute string CheckID;
};

//
interface CIM_OSVersionCheck : CIM_Check {
   attribute string MaximumVersion;
   attribute string MinimumVersion;
   attribute AMS_Util::AMS_uint16 TargetOperatingSystem;
};

//
interface CIM_ArchitectureCheck : CIM_Check {
   attribute AMS_Util::AMS_uint16 ArchitectureType;
};

//
interface CIM_DirectorySpecification : CIM_Check {
   attribute string DirectoryPath;
};

//
interface CIM_Action {
   // CIM_Action or AMS_ApplicationSpecification::AMS_ESESpec
   // or AMS_ApplicationSpecification::AMS_SoftwareFeatureSpec
   // or AMS_ApplicationDeploymentSpecification::AMS_DeploymentLinkSpec
   readonly attribute Object Owner;
   readonly attribute AMS_ApplicationSpecification::AMS_ActionCheckCase state;
   //
   CIM_ActionListIterator GetActionSequenceIterator ( );
   readonly attribute CIM_ActionList ActionSequence;
   attribute string ActionID;
   attribute AMS_Util::AMS_uint16 TargetOperatingSystem;
};

//
interface CIM_ExecuteProgram : CIM_Action {
   attribute string CommandLine;
   attribute string ProgramPath;
};

//
```

```
interface CIM_FileAction : CIM_Action {
};

//
interface CIM_CopyFileAction : CIM_FileAction {
   readonly attribute CIM_DirectorySpecification ToDirectoryAction;
   readonly attribute CIM_DirectorySpecification FromDirectorySpecification;
   attribute string Source;
   attribute string Destination;
};

//
interface CIM_RedundancyGroup : CIM_LogicalElement {
};

//
interface CIM_AdminDomain {
   attribute string Name;
};

//
interface CIM_StorageExtent : CIM_LogicalDevice {
};

//
interface CIM_LogicalDisk : CIM_StorageExtent {
};

//
interface CIM_Memory : CIM_LogicalDevice {
};

//
interface CIM_PowerSupply : CIM_LogicalDevice {
};

//
interface CIM_Watchdog : CIM_LogicalDevice {
};

//
interface CIM_Processor : CIM_LogicalDevice {
   attribute AMS_Util::AMS_uint16 LoadPercentage;
   attribute AMS_Util::AMS_uint16 CPUStatus;
};

//
interface CIM_LogicalPort : CIM_LogicalDevice {
};

//
interface CIM_NetworkPort : CIM_LogicalPort {
};

//
interface CIM_EthernetPort : CIM_NetworkPort {
};

//
interface CIM_Display : CIM_LogicalDevice {
```

```
};

//
interface CIM_Sensor : CIM_LogicalDevice {
};

//
interface CIM_PhysicalElement {
   CIM_LogicalDeviceListIterator GetRealizesIterator ( );
   readonly attribute CIM_LogicalDeviceList Realizes;
   readonly attribute CIM_Location PhysicalElementLocation;
};

//
interface CIM_Log : CIM_EnabledLogicalElement {
   attribute AMS_Util::AMS_uint64 MaxNumberOfRecords;
   attribute AMS_Util::AMS_uint64 CurrentNumberOfRecord;
   attribute string Name;
   AMS_Util::AMS_uint32 ClearLog ( )
      raises (AMS_Util::AMS_Error);
};

//
interface CIM_ManagedElement {
};

//
interface CIM_RecordForLog : CIM_ManagedElement {
   attribute string RecordFormat;
   attribute string RecordData;
   attribute string Locale;
};

//
interface CIM_Dependency {
};

//
interface CIM_ConnectivityCollection {
   CIM_ProtocolEndPointListIterator GetMemberOfCollectionIterator ( );
   readonly attribute CIM_ProtocolEndPointList MemberOfCollection;
   readonly attribute CIM_Network Owner;
   attribute AMS_Util::AMS_uint64 InstanceID;
};

//
interface CIM_EthernetPortStatistics {
   readonly attribute CIM_LANEndPoint Owner;
   attribute AMS_Util::AMS_uint64 InstanceID;
   attribute AMS_Util::AMS_uint64 BytesTransmitted;
   attribute AMS_Util::AMS_uint64 BytesReceived;
   attribute AMS_Util::AMS_uint64 PacketsTransmitted;
   attribute AMS_Util::AMS_uint64 PacketsReceived;
};

//
interface CIM_Network : CIM_AdminDomain {
   CIM_ConnectivityCollectionListIterator GetHostedCollectionIterator ( );
   readonly attribute CIM_ConnectivityCollectionList HostedCollection;
   CIM_NetworkListIterator GetSubnetComponentIterator ( );
```

```
      readonly attribute CIM_NetworkList SubnetComponent;
   };

   //
   interface CIM_NextHopIPRoute : CIM_NextHopRoute {
      attribute AMS_Util::AMS_uint16 RouteDerivation;
      attribute string DestinationMask;
      attribute AMS_Util::AMS_uint8 PrefixLength;
      attribute AMS_Util::AMS_uint16 AddressType;
   };

   //
   interface CIM_Process : CIM_EnabledLogicalElement {
      CIM_ThreadListIterator GetThreadListIterator ( );
      CIM_ThreadList ProcessThread;
      CIM_ServiceListIterator GetServiceListIterator ( );
      CIM_ServiceListServiceList;
      attribute string OSName;
      attribute string Handle;
   };

   //
   interface CIM_Thread : CIM_EnabledLogicalElement {
      attribute string OSName;
      attribute string Handle;
   };

   //
   interface CIM_UnixProcess : CIM_Process {
      attribute AMS_Util::AMS_uint64 ProcessGroupID;
      attribute AMS_Util::AMS_uint64 RealUserID;
      attribute AMS_Util::AMS_stringList Parameters;
   };

   //
   interface CIM_UnixThread : CIM_Thread {
   };


};

#endif/* _AMS_CIM_IDL_ */
```

8.6.9 AMS_LogicalHardware.idl

```
// Copyright 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
// Themis Computer and Progeny Systems Corporation.

#include "AMS_Util.idl"
#include "AMS_CIM.idl"

#ifndef _AMS_LogicalHardware_IDL_
#define _AMS_LogicalHardware_IDL_

#pragma prefix "omg.org"

#include "AMS_LogicalHardwareSpecification.idl"
#include "AMS_SupportedApplicationModel.idl"
```

```
module AMS_ApplicationDeployment {
   interface AMS_DeploymentLink;
   typedef sequence<AMS_DeploymentLink> AMS_DeploymentLinkList;
   interface AMS_DeploymentLinkListIterator;
};

module AMS_Client {
   interface AMS_IndicationSink;
} ;

module AMS_LogicalHardware {
   //
   enum AMS_StdHWUtilisation {
```

Issue 11404 - Non-coherent naming of some items in enumerates

```
      HU_NONSTD,
      HU_CPU_LOAD,
      HU_NETWORK_LOAD,
      HU_NETWORK_BANDWIDTH,
      HU_VIRTUAL_MEMORY,
      HU_VIRTUAL_MEMORY_OCCUPATION,
      HU_TOTAL_MEMORY,
      HU_TOTAL_MEMORY_OCCUPATION,
      HU_PROCESS_NUMBER,
      HU_THREAD_NUMBER,
      HU_DISK,
      HU_DISK_OCCUPATION
   }

   //
   const string AMS_Host_CLASSID
         = "IDL:omg.org/AMS_LogicalHardware/AMS_Host:1.0";
   interface AMS_Host ;
   typedef sequence<AMS_Host> AMS_HostList;
   interface AMS_HostListIterator : AMS_Util::AMS_Iterator {
      AMS_Host GetCurrentItem ()
         raises (AMS_Util::AMS_NoSuchElementException);
   };

   //
   const string AMS_Router_CLASSID
         = "IDL:omg.org/AMS_LogicalHardware/AMS_Router:1.0";
   interface AMS_Router ;
   typedef sequence<AMS_Router> AMS_RouterList;
   interface AMS_RouterListIterator : AMS_Util::AMS_Iterator {
      AMS_Router GetCurrentItem ()
         raises (AMS_Util::AMS_NoSuchElementException);
   };

   //
   const string AMS_Switch_CLASSID
         = "IDL:omg.org/AMS_LogicalHardware/AMS_Switch:1.0";
   interface AMS_Switch ;
   typedef sequence<AMS_Switch> AMS_SwitchList;
   interface AMS_SwitchListIterator : AMS_Util::AMS_Iterator {
      AMS_Switch GetCurrentItem ()
         raises (AMS_Util::AMS_NoSuchElementException);
   };
```

```
//
const string AMS_HardwareGroup_CLASSID
      = "IDL:omg.org/AMS_LogicalHardware/AMS_HardwareGroup:1.0";
interface AMS_HardwareGroup ;
typedef sequence<AMS_HardwareGroup> AMS_HardwareGroupList;
interface AMS_HardwareGroupListIterator : AMS_Util::AMS_Iterator {
   AMS_HardwareGroup GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_ComputerSystem_CLASSID
      = "IDL:omg.org/AMS_LogicalHardware/AMS_ComputerSystem:1.0";
interface AMS_ComputerSystem ;
typedef sequence<AMS_ComputerSystem> AMS_ComputerSystemList;
interface AMS_ComputerSystemListIterator : AMS_Util::AMS_Iterator {
   AMS_ComputerSystem GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_Printer_CLASSID
      = "IDL:omg.org/AMS_LogicalHardware/AMS_Printer:1.0";
interface AMS_Printer ;
typedef sequence<AMS_Printer> AMS_PrinterList;
interface AMS_PrinterListIterator : AMS_Util::AMS_Iterator {
   AMS_Printer GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_DomainManager_CLASSID
      = "IDL:omg.org/AMS_LogicalHardware/AMS_DomainManager:1.0";
interface AMS_DomainManager ;
typedef sequence<AMS_DomainManager> AMS_DomainManagerList;
interface AMS_DomainManagerListIterator : AMS_Util::AMS_Iterator {
   AMS_DomainManager GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_Domain_CLASSID
      = "IDL:omg.org/AMS_LogicalHardware/AMS_Domain:1.0";
interface AMS_Domain ;
typedef sequence<AMS_Domain> AMS_DomainList;
interface AMS_DomainListIterator : AMS_Util::AMS_Iterator {
   AMS_Domain GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_OperatingSystem_CLASSID
      = "IDL:omg.org/AMS_LogicalHardware/AMS_OperatingSystem:1.0";
interface AMS_OperatingSystem ;
typedef sequence<AMS_OperatingSystem> AMS_OperatingSystemList;
interface AMS_OperatingSystemListIterator : AMS_Util::AMS_Iterator {
   AMS_OperatingSystem GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};
```

```
   //
   const string AMS_LANEndPoint_CLASSID
         = "IDL:omg.org/AMS_LogicalHardware/AMS_LANEndPoint:1.0";
   interface AMS_LANEndPoint ;
   typedef sequence<AMS_LANEndPoint> AMS_LANEndPointList;
   interface AMS_LANEndPointListIterator : AMS_Util::AMS_Iterator {
      AMS_LANEndPoint GetCurrentItem ()
         raises (AMS_Util::AMS_NoSuchElementException);
   };

   //
   const string AMS_Property_StdHWUtilisation_CLASSID
         = "IDL:omg.org/AMS_LogicalHardware/AMS_Property_StdHWUtilisation:1.0";
   interface AMS_Property_StdHWUtilisation;
   typedef sequence<AMS_Property_StdHWUtilisation> AMS_Property_StdHWUtilisationList;
   interface AMS_Property_StdHWUtilisationListIterator : AMS_Util::AMS_Iterator {
      AMS_Property_StdHWUtilisation GetCurrentItem ()
         raises (AMS_Util::AMS_NoSuchElementException);
   };

   //
   struct AMS_RTHardwareUtilisation {
      AMS_Util::AMS_uint16 CPULoad;
      AMS_Util::AMS_uint16 MemoryLoad;
      AMS_Util::AMS_uint16 DskUsage;
   };

   //
   interface AMS_ComputerSystem : AMS_CIM::CIM_ComputerSystem {
      AMS_CIM::CIM_ServiceAccessPointListIterator GetHostedAccessPointIterator ( );
      readonly attribute AMS_CIM::CIM_ServiceAccessPointList HostedAccessPoint;
      // list of AMS_Domain and/or AMS_CIM::CIM_LogicalDevice and/or
AMS_HardwareGroupList
      AMS_Util::AMS_ObjectListIterator GetSystemComponentIterator ( );
      readonly attribute AMS_Util::AMS_ObjectList SystemComponent;
      //
      readonly attribute AMS_CIM::CIM_Location ElementLocation;
      AMS_OperatingSystemListIterator GetInstalledOSIterator ( );
      readonly attribute AMS_OperatingSystemList InstalledOS;
      AMS_OperatingSystemListIterator GetRunningOSIterator ( );
      readonly attribute AMS_OperatingSystemList RunningOS;
      readonly attribute
         AMS_LogicalHardwareSpecification::AMS_ConfigurationSpecification ConfSpecCS;
      attribute string Name;
      attribute string ArchitectureInfo;
      attribute AMS_Util::AMS_uint16 Status;
      attribute AMS_Util::AMS_uint16 NetworkLoad;
   };

   //
   interface AMS_Host : AMS_ComputerSystem {
      AMS_CIM::CIM_NextHopRouteListIterator GetHostedRouteIterator ( );
      readonly attribute AMS_CIM::CIM_NextHopRouteList HostedRoute;
      AMS_Property_StdHWUtilisationListIterator GetStdHWUtilisationListIterator ( );
      readonly attribute AMS_Property_StdHWUtilisationList RTHU;
      AMS_ApplicationDeployment::AMS_DeploymentLinkListIterator
                                             GetHostUsedIterator ( );
      readonly attribute AMS_ApplicationDeployment::AMS_DeploymentLinkList HostUsed;
   };
```

```
//
interface AMS_Router : AMS_ComputerSystem {
   AMS_CIM::CIM_NextHopRouteListIterator GetHostedRouteIterator ( );
   readonly attribute AMS_CIM::CIM_NextHopRouteList HostedRoute;
};

//
interface AMS_Switch : AMS_ComputerSystem {
};

//
interface AMS_HardwareGroup : AMS_CIM::CIM_AdminDomain {
   AMS_ComputerSystemListIterator GetSystemComponentIterator ( );
   readonly attribute AMS_ComputerSystemList SystemComponent;
   readonly attribute
      AMS_LogicalHardwareSpecification::AMS_ConfigurationSpecification ConfSpecHG;
   void SubscribeHWStatusChange ( in AMS_Client::AMS_IndicationSink sink,
                        out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void SubscribeHWStatus ( in AMS_Client::AMS_IndicationSink sink,
                    in AMS_Util::AMS_uint16 delay,
                    out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   AMS_Util::AMS_uint16 GetMergedStatus ( )
      raises (AMS_Util::AMS_Error);
   void SubscribeMergedHWStatusChange ( in AMS_Client::AMS_IndicationSink sink,
                           out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void SubscribeMergedHWStatus ( in AMS_Client::AMS_IndicationSink sink,
                           in AMS_Util::AMS_uint16 delay,
                           out AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
   void Unsubscribe ( in AMS_Util::AMS_uint32 subscriptionID )
      raises (AMS_Util::AMS_Error);
};

//
interface AMS_Printer : AMS_ComputerSystem {
};

//
interface AMS_DomainManager : AMS_Host {
   readonly attribute AMS_Domain DomainManagerRole;
};

//
interface AMS_Domain : AMS_CIM::CIM_AdminDomain {
   AMS_DomainManagerListIterator GetDomainManagerRoleIterator ( );
   readonly attribute AMS_DomainManagerList DomainManagerRole;
   AMS_ComputerSystemListIterator GetSystemComponentIterator ( );
   readonly attribute AMS_ComputerSystemList SystemComponent;
   readonly attribute
      AMS_LogicalHardwareSpecification::AMS_ConfigurationSpecification
                                                              ConfSpecDom;
};

//
interface AMS_OperatingSystem : AMS_CIM::CIM_OperatingSystem {
   readonly attribute AMS_ComputerSystem Owner;
   AMS_SAM::AMS_SupportedApplicationModelListIterator
```

```
                                                GetAMSupportedByOSIterator ( );
      readonly attribute
         AMS_SAM::AMS_SupportedApplicationModelList AMSupportedByOS;
      attribute AMS_SAM::AMS_OSType OSType;
      readonly attribute
         AMS_LogicalHardwareSpecification::AMS_ConfigurationSpecification ConfSpecOS;
      attribute string Name;
      attribute string Version;
   };

   //
   interface AMS_LANEndPoint : AMS_CIM::CIM_LANEndPoint {
      attribute AMS_Util::AMS_uint16 Status;
      attribute AMS_Util::AMS_uint16 NetworkLoad;
   };

   //
   interface AMS_Property_StdHWUtilisation {
      attribute string Name;
      attribute string Value;
      attribute string InstanceID;
      attribute AMS_StdHWUtilisation StdName;
   };

};

#endif/* _AMS_LogicalHardware_IDL_ */
```

8.6.10 AMS_LogicalHardwareSpecification.idl

```
// Copyright 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
// Themis Computer and Progeny Systems Corporation.

#include "AMS_Util.idl"
#include "AMS_CIM.idl"

#ifndef _AMS_LogicalHardwareSpecification_IDL_
#define _AMS_LogicalHardwareSpecification_IDL_

#pragma prefix "omg.org"

module AMS_ApplicationDeploymentSpecification {
   interface AMS_DeploymentLinkSpec;
};

module AMS_LogicalHardware {
   interface AMS_ComputerSystem;
   interface AMS_OperatingSystem;
   interface AMS_Domain;
   interface AMS_HardwareGroup;
};

module AMS_LogicalHardwareSpecification {
   //
   enum AMS_CoupleName {
      CS_NOTNORMALIZED,
      CS_NAME,
      CS_FRU,
      CS_POSITION,
      CS_INTERFACE,
```

```
   CS_MFGDATETIME,
   CS_MANUFACTURER,
   CS_PRODUCTNAME,
   CS_PRODUCTVERSION,
   CS_SERIALNUMBER,
   CS_PRODUCTTYPE,
   CS_ASSETTAG,
   CS_CHASSISTYPE,
   CS_MACADDRESS,
   CS_POWERSATE,
   CS_STATUS,
   CS_POSTRESULT
};
typedef sequence<AMS_CoupleName> AMS_CoupleNameList;
//
const string AMS_CodedConstraint_CLASSID
      = "IDL:omg.org/AMS_LogicalHardwareSpecification/AMS_CodedConstraint:1.0";
interface AMS_CodedConstraint ;
typedef sequence<AMS_CodedConstraint> AMS_CodedConstraintList;
interface AMS_CodedConstraintListIterator : AMS_Util::AMS_Iterator {
   AMS_CodedConstraint GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_ConfigurationSpecification_CLASSID
= "IDL:omg.org/AMS_LogicalHardwareSpecification/AMS_ConfigurationSpecification:1.0";
interface AMS_ConfigurationSpecification ;
typedef sequence<AMS_ConfigurationSpecification> AMS_ConfigurationSpecificationList;
interface AMS_ConfigurationSpecificationListIterator : AMS_Util::AMS_Iterator {
   AMS_ConfigurationSpecification GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_NameValueCouple_CLASSID
      = "IDL:omg.org/AMS_LogicalHardwareSpecification/AMS_NameValueCouple:1.0";
interface AMS_NameValueCouple ;
typedef sequence<AMS_NameValueCouple> AMS_NameValueCoupleList;
interface AMS_NameValueCoupleListIterator : AMS_Util::AMS_Iterator {
   AMS_NameValueCouple GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_RangeConstraint_CLASSID
      = "IDL:omg.org/AMS_LogicalHardwareSpecification/AMS_RangeConstraint:1.0";
interface AMS_RangeConstraint ;
typedef sequence<AMS_RangeConstraint> AMS_RangeConstraintList;
interface AMS_RangeConstraintListIterator : AMS_Util::AMS_Iterator {
   AMS_RangeConstraint GetCurrentItem ()
      raises (AMS_Util::AMS_NoSuchElementException);
};

//
const string AMS_SetConstraint_CLASSID
      = "IDL:omg.org/AMS_LogicalHardwareSpecification/AMS_SetConstraint:1.0";
interface AMS_SetConstraint ;
typedef sequence<AMS_SetConstraint> AMS_SetConstraintList;
interface AMS_SetConstraintListIterator : AMS_Util::AMS_Iterator {
```

```
      AMS_SetConstraint GetCurrentItem ()
         raises (AMS_Util::AMS_NoSuchElementException);
   };

   //
   const string AMS_ValueConstraint_CLASSID
         = "IDL:omg.org/AMS_LogicalHardwareSpecification/AMS_ValueConstraint:1.0";
   interface AMS_ValueConstraint ;
   typedef sequence<AMS_ValueConstraint> AMS_ValueConstraintList;
   interface AMS_ValueConstraintListIterator : AMS_Util::AMS_Iterator {
      AMS_ValueConstraint GetCurrentItem ()
         raises (AMS_Util::AMS_NoSuchElementException);
   };

   //
   interface AMS_ConfigurationSpecification {
      readonly attribute AMS_LogicalHardware::AMS_ComputerSystem ConfSpecCS;
      readonly attribute AMS_LogicalHardware::AMS_OperatingSystem ConfSpecOS;
      readonly attribute AMS_LogicalHardware::AMS_Domain ConfSpecDom;
      readonly attribute AMS_LogicalHardware::AMS_HardwareGroup ConfSpecHG;
      readonly attribute
         AMS_ApplicationDeploymentSpecification::AMS_DeploymentLinkSpec ConfSpecDLS;
      AMS_NameValueCoupleListIterator GetNameValueIterator ( );
      readonly attribute AMS_NameValueCoupleList NameValue;
      attribute string InstanceID;
   };

   //
   interface AMS_NameValueCouple {
      readonly attribute AMS_ConfigurationSpecification Owner;
      attribute AMS_CoupleName StdName;
      readonly attribute AMS_ValueConstraint Constraint;
      attribute string InstanceID;
      attribute string Name;
   };

   //
   interface AMS_ValueConstraint {
      readonly attribute AMS_NameValueCouple Owner;
      attribute string InstanceID;
   };

   //
   interface AMS_CodedConstraint : AMS_ValueConstraint {
      attribute string constraint;
   };

   //
   interface AMS_RangeConstraint : AMS_ValueConstraint {
      attribute string from;
      attribute string to;
   };

   //
   interface AMS_SetConstraint : AMS_ValueConstraint {
      attribute string set;
   };

};
```

```
#endif/* _AMS_LogicalHardwareSpecification_IDL_ */
```

8.6.11 AMS_SupportedApplicationModel.idl

```
// Copyright 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
// Themis Computer and Progeny Systems Corporation.

#include "AMS_Util.idl"
#include "AMS_CIM.idl"

#ifndef _AMS_SupportedApplicationModel_IDL_
#define _AMS_SupportedApplicationModel_IDL_

#pragma prefix "omg.org"

module AMS_SAM {
   //
   enum AMS_OSType {
      Unknown,
      Other,
      MACOS,
      ATTUNIX,
      DGUX,
      DECNT,
      Tru64UNIX,
      OpenVMS,
      HPUX,
      AIX,
      MVS,
      OS400,
      OS_2,
      JavaVM,
      MSDOS,
      WIN3x,
      WIN95,
      WIN98,
      WINNT,
      WINCE,
      NCR3000,
      NetWare,
      OSF,
      DC_OS,
      ReliantUNIX,
      SCOUnixWare,
      SCOOpenServer,
      Sequent,
      IRIX,
      Solaris,
      SunOS,
      U6000,
      ASERIES,
      TandemNSK,
      TandemNT,
      BS2000,
      LINUX,
      Lynx,
      XENIX,
      VM,
      InteractiveUNIX,
      BSDUNIX,
```

```
   FreeBSD,
   NetBSD,
   GNUHurd,
   OS9,
   MACHKernel,
   Inferno,
   QNX,
   EPOC,
   IxWorks,
   VxWorks,
   MiNT,
   BeOS,
   HPMPE,
   NextStep,
   PalmPilot,
   Rhapsody,
   Windows2000,
   Dedicated,
   OS_390,
   VSE,
   TPF,
   Windows_R_Me,
   CalderaOpenUNIX,
   OpenBSD,
   NotApplicable,
   WindowsXP,
   z_OS,
```

Issue 11519 - Vista

```
   WindowsVista
};
typedef sequence<AMS_OSType> AMS_OSTypeList;
//
enum AMS_ModelType {
   AMS_PROCESS,
   AMS_J2EE,
   AMS_CCM
};
typedef sequence<AMS_ModelType> AMS_ModelTypeList;
//
enum AMS_Control {
   AMS_LOAD,
   AMS_LOAD_START,
   AMS_START,
   AMS_STOP,
   AMS_HALT,
   AMS_CONTINUE,
   AMS_SHUTDOWN,
   AMS_RECOVER,
   AMS_UNLOAD,
   AMS_LOAD_DIRTY,
   LOAD_START_DIRTY,
   AMS_STOP_HALTED,
   AMS_RECLAIM,
   AMS_ALLOCATE,
   AMS_RECOVER_DIRTY
};
typedef sequence<AMS_Control> AMS_ControlList;
//
enum AMS_State {
```

```
      AMS_EXECUTABLE,
      AMS_HALTED,
      AMS_LOADED,
      AMS_RUNNING,
      AMS_STOPPED,
      AMS_UNALLOCATED,
      AMS_ERROR
   };
   typedef sequence<AMS_State> AMS_StateList;
   //
   enum AMS_StdMechanism {
      MS_NONSTD,
      MS_POSIXSIGNAL
   };
   typedef sequence<AMS_StdMechanism> AMS_StdMechanismList;
   //
   const string AMS_SupportedApplicationModel_CLASSID
      = "IDL:omg.org/AMS_SupportedApplicationModel/AMS_SupportedApplicationModel:1.0";
   interface AMS_SupportedApplicationModel ;
   typedef sequence<AMS_SupportedApplicationModel> AMS_SupportedApplicationModelList;
   interface AMS_SupportedApplicationModelListIterator : AMS_Util::AMS_Iterator {
      AMS_SupportedApplicationModel GetCurrentItem ()
         raises (AMS_Util::AMS_NoSuchElementException);
   };

   //
   const string AMS_Mechanism_CLASSID
         = "IDL:omg.org/AMS_SupportedApplicationModel/AMS_Mechanism:1.0";
   interface AMS_Mechanism;
   typedef sequence<AMS_Mechanism> AMS_MechanismList;
   interface AMS_MechanismListIterator : AMS_Util::AMS_Iterator {
      AMS_Mechanism GetCurrentItem ()
         raises (AMS_Util::AMS_NoSuchElementException);
   };

   //
   interface AMS_SupportedApplicationModel : AMS_CIM::CIM_LogicalElement {
      attribute AMS_ModelTypeList SupportedModelType;
      attribute AMS_OSTypeList SupportedOSType;
      attribute AMS_ControlList SupportedControls;
      attribute AMS_StateList SupportedStates;
      attribute string Name;
      attribute string ConfigurationInfo;
      AMS_MechanismListIterator GetSupportedMechanismsIteratorOnControl ( in
AMS_Control control );
      AMS_MechanismList GetSupportedMechanismsOnControl ( in AMS_Control control );
   };

   //
   interface AMS_Mechanism {
      attribute string Name;
      attribute string Value;
      attribute string InstanceID;
      attribute AMS_StdMechanism StdName;
   };

};

#endif/* _AMS_SupportedApplicationModel_IDL_ */
```

9 XML Platform Specific Model

9.1 Mapping Rationale

9.1.1 Objective

The objective of this PSM is to normalize the format of some of the files which can be read or written by an AMSM service. The uses of these files by an AMSM service are threefold:

- Firstly, these files may be the configuration files allowing the user of the AMSM service (integrator…) to initialize the service with software system specifications, application specifications, deployment specifications, and a (first) drawing of the network.
- Secondly, these files may be used as a backup capability allowing an AMSM service to be re-started with its previously recorded state.
- Lastly, these files may also be used so as to exchange data amongst multiple instantiations of the AMSM service. These data might also be exchanged directly (i.e., not through a file, in a future extension of this norm).

These uses require some specific features for the XML PSM:

- (R1) These files have to be readable by human operators. Even if their automated generation is expected, it will be useful for a human operator (such as a system integrator for instance) to be able to read and correct the AMSM service configuration files. The use of XML for their format, allows *per se* these files to be readable by human operators. Yet, this requirement asks to avoid tedious perusal of large files when seeking for the data of a logical object. So, it implies to collect the descriptions of logically linked objects in a single area of a file. For instance, it must be possible to gather in one XML structure the specification of an application with its related Executable Software Elements.
- (R2) All classes which contain persistent data have to be stored in these files. This implies that this PSM is not limited to “Specification” classes but have been extended to almost all the classes of the PIM.
- (R3) It must also be possible to state basic alterations of elements: applications, hosts, networks, and so in a such way, for instance, to exchange them amongst AMSM services. This requirement implies the possibility to define sub-elements (i.e., objects which belong to other objects) at the XML root level and to be able to reference the owner of such an element.

9.1.2 Mapping principle

The XML format is described with XML Schema files (cf. [XMLSchema]). The rules used to map the PIM into the XML Schema files are:

- Each package of the PIM is mapped to an XML Schema file.

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"xmlns:amsm="http://www.omg.org/
amsm"
   targetNamespace="http://www.omg.org/amsm"
   xmlns="http://www.omg.org/amsm">
      ...
</ xsd:schema>
```

- Each package of the PIM includes the XML Schema files of the packages on which it depends (cf. 7.1.1 – Figure 2).
- Data types (uint16, date) are directly mapped to XML Schema types using the rules explained in 9.3.

- Enumeration classes are mapped to XML Schema simple types defined as restriction based on “xsd:string:”

```
<xsd:simpleType name="enumeration-name">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="an-item"/>
      ...
   </xsd:restriction>
</xsd:simpleType>
```

- Non enumeration classes are mapped to XML Schema complex types whose name is the concatenation of the name of the class and the string “Type:”

```
<xsd:complexType name="class-nameType">
   ...
</xsd:complexType>
```

- If a non enumeration class has a super class, the preceding mapping is completed with an XML Schema extension definition:

```
   <xsd:complexContent>
   <xsd:extension base="super-class-nameType">
      ...
   </xsd:extension>
   </xsd:complexContent>
```

- AMS_Property<C,V> template is mapped as follows:
- A complex type whose name is AMS_Property followed by an "_" and the name used for the first parameter in the template spec. (without "AMS") plus a trailing "Type;" this complex type will also contain an attribute named "StdName" whose type is a simple Type named as the first parameter of the template; i.e., in the case of AMS_Property<AMS_StdState,ST_NONSTD>:

```
<xsd:complexType name="AMS_Property_StdStateType">
   <xsd:attribute name="Name" type="xsd:string" use="required"/>
   <xsd:attribute name="Value" type="xsd:string" use="required"/>
   <xsd:attribute name="InstanceID" type="xsd:string" use="required"/>
   <xsd:attribute name="StdName" type="AMS_StdState" use="required"/>
</xsd:complexType>

<xsd:simpleType name="AMS_StdState">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="ST_NONSTD"/>
      <xsd:enumeration value="ST_ENV"/>
   </xsd:restriction>
</xsd:simpleType>
```

- Attributes of non-enumeration classes are next mapped with required XML Schema attributes as follow:

```
   <xsd:attribute name="attribute-name" type="attribute-type-map" use="required"/>
```

- Associations of non-enumeration classes are next mapped as follow:
- Compositions with multiplicity equal to 1 are mapped to a required XML Schema attributes whose name is the name of the association and type is the type of the target class of the composition (which is, in the AMSM PIM, always an enumeration class).

```
   <xsd:attribute name="assoc-name" type="class-name" use="required"/>
```

- Compositions with multiplicity greater than 1 are mapped to an XML Schema sequence of elements whose name is the name of the association and type is the type of the target class of the composition (which is, in the AMSM PIM, always an enumeration class).

```
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:element name="assoc-name" type="class-name"/>
   </xsd:sequence>
```

- Aggregations are mapped to XML Schema sequences with elements whose name is the name of the association and type is the type of the aggregation (cf. further for the definition of this aggregation type). The minimum and maximum occurrences of these sequences are those expressed in the PIM on the association.

```
<xsd:sequence>
   <xsd:element name="assoc-name" type="assoc-type" minOccurs="n" maxOccurs="m"/>
</xsd:sequence>
```

- Simple associations are mapped to XML Schema sequences with elements whose name is the name of the association and type is AMS_Ref (cf. further for the definition of this type). The minimum and maximum occurrences of these sequences are those expressed in the PIM on the association.

```
<xsd:sequence>
   <xsd:element name="assoc-name" type="assoc-type" minOccurs="n" maxOccurs="m"/>
</xsd:sequence>
```

- Associations whose reverse association is an aggregation or a composition are not mapped.
- The AMS_Ref type is defined as a string which contains the name of the associated element (cf. naming convention in Section 7.1.2.1):

```
<xsd:complexType name="AMS_Ref">
   <xsd:attribute name="AMS_ID" type="xsd:string" use="required"/>
</xsd:complexType>
```

- Each aggregation goes with a so-called “aggregation type” which is defined as a sequence of choices:
- Either elements whose name is the name of the target class and type is the type of the target class of the aggregation,
- Or elements whose name is the name of the target class plus “Ref” and type is AMS_Ref (i.e., a string which contains the name of the aggregated element).
- The first choice is designed to describe complete objects (cf. R1 in 9.1.1.), the second one is designed to deal with objects which are separated of their container (cf. R3 in 9.1.1).
- The name of the aggregation type is the name of the association plus “_LinkType:”

```
<xsd:complexType name="assoc-name_LinkType">
   <xsd:sequence minOccurs="n" maxOccurs="m">
      <xsd:choice>
         <xsd:element name="target-class-name" type="target-classType"/>
         <xsd:element name="target-class-nameRef" type="AMS_Ref"/>
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>
```

- After defining XML Schema types, the effective XML structure is derived by listing the elements which are deemed as appropriate for the package: all these elements will be allowed to appear directly under the root level of a file.
- For each of the types corresponding to these “root” elements, an optional attribute is added which contains the name (cf. naming convention in Section 7.1.2.1) of the owner element. This attribute may be used in the case of the description of objects which are separated of their container (cf. R3 in 9.1.1.).

```
<xsd:attribute name="OwnerId" type="xsd:string"/>
```

These rules are not intended to be general rules to map an UML PIM to a XML Schema PSM. They are specific rules established for the AMSM specific case.

9.1.3 Mapping exceptions

The preceding rules deal neither with all the cases which arises in the AMSM PIM, nor with some of the requirements of XML Schemas.

Hereby are those specific cases:

- A specific XML schema file is added to define the type AMS_Ref.
- The association CIM_SystemComponent appears two or three times on some classes (ASM_SoftwareSystem, ASM_Application and ASM_ComputerSystem), which leads to multiple sequences and types with the same name. Since this is not allowed in an XML Schema:
- In the case of an aggregation, these associations are gathered in one sequence with one aggregation type (whose name is the name of the association plus the name the class plus "_LinkType").
- In the case of a simple association, they are gathered in one sequence of type AMS_Ref.
- For instance:

```
<xsd:complexType name="CIM_SystemComponent_Application_LinkType">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="AMS_ExecutableSoftwareElement"
                      type="AMS_ExecutableSoftwareElementType"/>
         <xsd:element name="AMS_ExecutableSoftwareElementRef"
                      type="AMS_Ref"/>
         <xsd:element name="AMS_RedundancyGroup"
                      type="AMS_RedundancyGroupType"/>
         <xsd:element name="AMS_RedundancyGroupRef"
                      type="AMS_Ref"/>
         <xsd:element name="AMS_LoadBalancingGroup"
                      type="AMS_LoadBalancingGroupType"/>
         <xsd:element name="AMS_LoadBalancingGroupRef"
                      type="AMS_Ref"/>
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>


<xsd:complexType name="AMS_ApplicationType">
   <xsd:complexContent>
   <xsd:extension base="CIM_ApplicationSystemType">
      <xsd:sequence>
         <xsd:element name="CIM_SystemComponent"
            type="CIM_SystemComponent_Application_LinkType"
            minOccurs="0" maxOccurs="unbounded"/>
         ...
      </xsd:sequence>
         ...
   </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

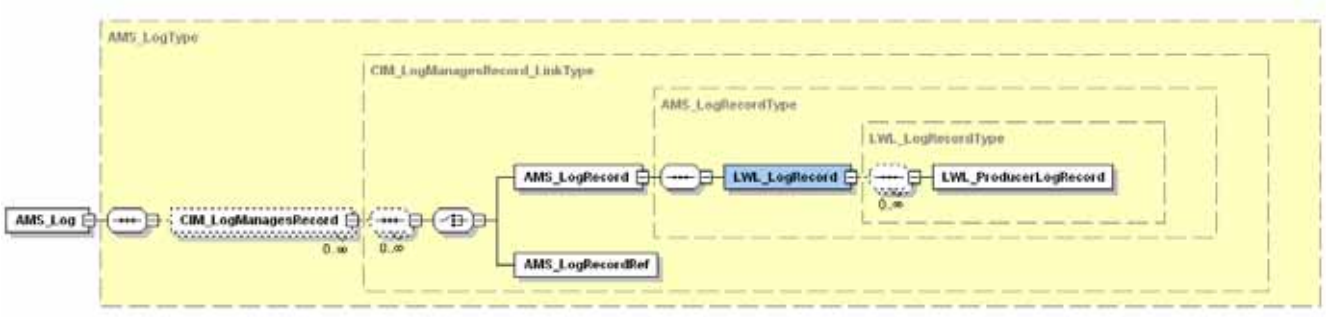
- Association classes are not yet properly mapped. The association classes of the PIM are: the AMS_ActionCheckLink and subclasses used by AMS_SoftwareFeature, AMS_ESESpec, AMS_DeploymentSpec, the AMS_SEShutdownDependency, AMS_SEStartDependency and subclasses. These associations and aggregations are mapped in complex types with name "_LinkType" and a design following the general model of an aggregation type but with the addition of either an attribute which has the AMS_ActionCheckCase (AMS_Action-CheckLink and subclasses) as a type, or an element or choice of elements which has an object of the class of the dependency (AMS_SEShutdownDependency, AMS_SEStartDependency and subclasses) as a type. For instance:

```
<xsd:complexType name="AMS_SoftwareElementAction_LinkType">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="CIM_Action" type="CIM_ActionType"/>
```

- Associations which lead to a target class which have subclasses (for instance: ASM_SoftwareElementCheck leads to CIM_Check which have four subclasses) are specified by giving all the possible target classes, i.e., the subclasses. For instance ASM_SoftwareElementCheck gives:

```
<xsd:complexType name="AMS_SoftwareElementCheck_LinkType">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="CIM_Check" type="CIM_CheckType"/>
         <xsd:element name="AMS_ApplicationModelCheck"
                      type="AMS_ApplicationModelCheckType"/>
         <xsd:element name="CIM_OSVersionCheck" type="CIM_OSVersionCheckType"/>
         <xsd:element name="CIM_ArchitectureCheck"
                      type="CIM_ArchitectureCheckType"/>
         <xsd:element name="CIM_CheckRef" type="AMS_Ref"/>
      </xsd:choice>
   </xsd:sequence>
   <xsd:attribute name="Case" type="AMS_ActionCheckCase" use="required"/>
</xsd:complexType>
```

- The configuration of the log part of the service is simplified, hand-written and gathered in the AMS_Management package with the following structure:



- Id attributes (CheckId) are moved to optional.
- IPv4Address, IPv6Address, SubnetMask and PrefixLength attributes of CIM_IP-ProtocolEndPointType are moved to optional.

9.1.4 Samples

9.1.4.1 Software System Specification

Following is an example of a specification of a system called “The System” owning two ESE called “A” and “B.”

A starts before B and stops after B.

A has a check before starting about the type and the version of the OS (OS number 67: Windows XP and version between 1 and 1024) on which it runs. It has also an action to start (that’s should be a must-have!): the complete command line will be “run_that_A –test.”

B is also defined with a check on the OS and an action to start: “run_that_B.”

```
<?xml version="1.0" encoding="UTF-8"?>
<amsm:AMS_ApplicationSpecification
   xmlns:amsm="http://www.omg.org/amsm"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.omg.org/amsm AMS_ApplicationSpecification.xsd">

   <AMS_SoftwareFeatureSpec TypeOfFeature="SYSTEM" Name="The System">

      <CIM_SoftwareFeatureSoftwareElement>
         <AMS_ESESpec ModelType="AMS_PROCESS" RedInitState="REDSTATE_NORG" Name="A">
            <AMS_SoftwareElementCheck Case="CASE_PRE_START">
               <CIM_OSVersionCheck
                  MaximumVersion="1024"
                  MinimumVersion="1"
```

```
                        TargetOperatingSystem="67"/>
                  </AMS_SoftwareElementCheck>
                  <AMS_SoftwareElementAction Case="CASE_START">
                     <AMS_ExecuteProgram
                        TargetOperatingSystem="67"
                        CommandLine="run_that_A -test"
                        ProgramPath="./"
                        Environment=""/>
                  </AMS_SoftwareElementAction>
                  <AMS_SEShutdownDependency>
                     <Ref AMS_ID="B"/>
                     <AMS_SEShutdownDependency TimeSinceShutdown="10"/>
                  </AMS_SEShutdownDependency>
               </AMS_ESESpec>
            </CIM_SoftwareFeatureSoftwareElement>

            <CIM_SoftwareFeatureSoftwareElement>
               <AMS_ESESpec ModelType="AMS_PROCESS" RedInitState="REDSTATE_NORG" Name="B">
                  <AMS_SoftwareElementCheck Case="CASE_PRE_START">
                     <CIM_OSVersionCheck
                        MaximumVersion="1024"
                        MinimumVersion="1"
                        TargetOperatingSystem="67"/>
                  </AMS_SoftwareElementCheck>
                  <AMS_SoftwareElementAction Case="CASE_START">
                     <AMS_ExecuteProgram
                        TargetOperatingSystem="67"
                        CommandLine="run_that_B"
                        ProgramPath="./"
                        Environment=""/>
                  </AMS_SoftwareElementAction>
                  <AMS_SEStartDependency>
                     <Ref AMS_ID="A"/>
                     <AMS_SEStartTimeDependency TimeSinceStartup="10"/>
                  </AMS_SEStartDependency>
               </AMS_ESESpec>
            </CIM_SoftwareFeatureSoftwareElement>

         </AMS_SoftwareFeatureSpec>

      </amsm:AMS_ApplicationSpecification>
```

9.1.5 Host

This second example presents a host named “Cinderella” with IP address “192.163.1.1”, a processor, a display, located in “CIV B-35”, with an installed and running Windows XP and some hosted routes.

Please note that the hosted routes are not defined in the structure but are defined by reference and, so, are supposed to be known.

```
<?xml version="1.0" encoding="UTF-8"?>
<amsm:AMS_LogicalHardware
   xmlns:amsm="http://www.omg.org/amsm"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.omg.org/amsm AMS_LogicalHardware.xsd">

   <CIM_Location Name="CIV B-35"/>

   <AMS_Host Name="Cinderella" ArchitectureInfo="String" Status="0" NetworkLoad="0">
      <CIM_HostedAccessPoint>
```

```
            <CIM_IPProtocolEndPoint Name="lan0" IPv4Address="192.163.1.1"/>
         </CIM_HostedAccessPoint>
         <CIM_SystemComponent>
            <CIM_Processor LoadPercentage="0" CPUStatus="0"/>
            <CIM_Display/>
         </CIM_SystemComponent>
         <CIM_ElementLocation AMS_ID="CIV B-35"/>
         <CIM_InstalledOS>
            <AMS_OperatingSystem OSType="Windows XP" Name="-" Version="1"/>
         </CIM_InstalledOS>
         <CIM_RunningOS>
            <AMS_OperatingSystem OSType="Windows XP" Name="-" Version="1"/>
         </CIM_RunningOS>
         <CIM_HostedRoute>
            <CIM_NextHopRouteRef AMS_ID="route-name"/>
            <CIM_NextHopRouteRef AMS_ID="other-route-name"/>
         </CIM_HostedRoute>
         <AMS_RTHU>
            <AMS_RTHardwareUtilisation CPULoad="0" MemoryLoad="0" DskUsage="0"/>
         </AMS_RTHU>
      </AMS_Host>

   </amsm:AMS_LogicalHardware>
```

9.2 Specific attributes and parameters information

In this paragraph, some attributes of class and parameters of operation roughly defined in the PIM are more specified in the context of the XML PSM.

Table 9.1 - Specific Attributes for XML PSM

Attribute	Comment
AMS_HWFilter:filter	Implementation dependant
AMS_SWFilter:filter	Implementation dependant
AMS_SupportedApplicationModel: ConfigurationInfo	Implementation dependant
AMS_ExecuteProgram:Environment	Implementation dependant
AMS_ExecuteProgram:CommandLine	Implementation dependant
AMS_ExecuteProgram:ProgramPath	Implementation dependant

Table 9.2 - Specific Parameters for XML PSM

Parameter	Comment
CreateHardwareGroup: connectivity	N/A
CreateHardwareGroup: devices	N/A
CreateHardwareGroup: resources	N/A

9.3 Specific Data Types

This paragraph specifies the data types in the context of the XML PSM.

Table 9.3 - Data types for XML PSM

Data type	Definition	Comment
Collection<Type>	N/A	Since, these data types are encountered in parameters of methods, they are not mapped to this PSM
datetime	xsd:string	
String	xsd:string	
uint8	xsd:short	
uint16	xsd:int	
uint32	xsd:long	
uint64	xsd:integer	
boolean	xsd:boolean	

9.4 Specific Failure Codes

N/A.

9.5 Conformance Criteria

This PSM deems all PIM data as mandatory and does not acknowledge any optional profile. This means that all the data of the PIM (including those stereotyped as “optional”) may be present in a compliant AMSM XML file.

This criterion does not imply that an implementation which conforms to the XML PSM must also have a PIM which conforms to all the compliance profiles. Yet it implies that an implementation of the XML PSM must not refuse as erroneous a file which contains data which are not recognized in the conformance criteria of its PIM.

This criterion is aimed to allow easier exchanges of configuration files amongst AMSM implementations.

9.6 Mapping

9.6.1 AMSUtil

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:amsm="http://www.omg.org/amsm"
   targetNamespace="http://www.omg.org/amsm"
   xmlns="http://www.omg.org/amsm">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         file AMS_Util.xsd
```

```
            XML Schema Definition for namespace AMS Management
            Copyright (c) 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
                  Themis Computer and ProgenySystems Corporation
         </xsd:documentation>
      </xsd:annotation>

      <xsd:complexType name="AMS_Ref">
         <xsd:attribute name="AMS_ID" type="xsd:string" use="required" />
      </xsd:complexType>

   </xsd:schema>
```

9.6.2 AMS Management

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:amsm="http://www.omg.org/amsm"
   targetNamespace="http://www.omg.org/amsm"
   xmlns="http://www.omg.org/amsm">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         file AMS_AMSManagement.xsd
         XML Schema Definition for namespace AMS Management
         Copyright (c) 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
               Themis Computer and ProgenySystems Corporation
      </xsd:documentation>
   </xsd:annotation>
   <xsd:include schemaLocation="AMS_Util.xsd" />
   <xsd:include schemaLocation="AMS_CIM.xsd" />
   <xsd:include schemaLocation="AMS_Application.xsd" />
   <xsd:include schemaLocation="AMS_ApplicationDeployment.xsd" />
   <xsd:include
      schemaLocation="AMS_ApplicationDeploymentSpecification.xsd" />
   <xsd:include schemaLocation="AMS_ApplicationSpecification.xsd" />
   <xsd:include schemaLocation="AMS_LogicalHardware.xsd" />
   <xsd:include schemaLocation="AMS_LogicalHardwareSpecification.xsd" />
   <xsd:include schemaLocation="AMS_SupportedApplicationModel.xsd" />
   <!-- types definition !-->
   <xsd:complexType name="LWL_LogRecordType">
      <xsd:sequence minOccurs="0" maxOccurs="unbounded">
         <xsd:element name="LWL_ProducerLogRecord"
            type="LWL_ProducerLogRecordType" />
      </xsd:sequence>
      <xsd:attribute name="id" type="xsd:integer" use="required" />
      <xsd:attribute name="time" type="xsd:string" use="required" />
   </xsd:complexType>
   <xsd:complexType name="LWL_ProducerLogRecordType">
      <xsd:attribute name="producerId" type="xsd:string"
         use="required" />
      <xsd:attribute name="producerName" type="xsd:string"
         use="required" />
      <xsd:attribute name="level" type="xsd:int" use="required" />
      <xsd:attribute name="logData" type="xsd:string" use="required" />
   </xsd:complexType>

   <xsd:complexType name="CIM_LogManagesRecord_LinkType"
      mixed="false">
      <xsd:sequence minOccurs="0" maxOccurs="unbounded">
         <xsd:choice>
            <xsd:element name="AMS_LogRecord"
               type="AMS_LogRecordType" />
```

```
            <xsd:element name="AMS_LogRecordRef" type="AMS_Ref" />
         </xsd:choice>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="AMS_LogRecordType">
      <xsd:complexContent>
         <xsd:extension base="CIM_RecordForLogType">
            <xsd:sequence minOccurs="1" maxOccurs="1">
               <xsd:element name="LWL_LogRecord"
                  type="LWL_LogRecordType" />
            </xsd:sequence>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <xsd:complexType name="AMS_LogType">
      <xsd:complexContent>
         <xsd:extension base="CIM_LogType">
            <xsd:sequence>
               <xsd:element name="CIM_LogManagesRecord"
                  type="CIM_LogManagesRecord_LinkType" minOccurs="0"
                  maxOccurs="unbounded" />
            </xsd:sequence>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <!-- structure definition !-->
   <xsd:element name="AMS_Log" type="AMS_LogType" />
</xsd:schema>
```

9.6.3 Application

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:amsm="http://www.omg.org/amsm"
   targetNamespace="http://www.omg.org/amsm"
   xmlns="http://www.omg.org/amsm">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         file AMS_Application.xsd
         XML Schema Definition for namespace AMS Management
         Copyright (c) 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
               Themis Computer and ProgenySystems Corporation
      </xsd:documentation>
   </xsd:annotation>
   <xsd:include schemaLocation="AMS_Util.xsd" />
   <xsd:include schemaLocation="AMS_CIM.xsd" />
   <xsd:include schemaLocation="AMS_SupportedApplicationModel.xsd" />
   <xsd:include schemaLocation="AMS_ApplicationSpecification.xsd" />
   <!-- types definition !-->
   <xsd:complexType name="CIM_SystemComponent_Application_LinkType"
      mixed="false">
      <xsd:sequence minOccurs="0" maxOccurs="unbounded">
         <xsd:choice>
            <xsd:element name="AMS_ExecutableSoftwareElement"
               type="AMS_ExecutableSoftwareElementType" />
            <xsd:element name="AMS_ExecutableSoftwareElementRef"
               type="AMS_Ref" />
```

```
            <xsd:element name="AMS_RedundancyGroup"
               type="AMS_RedundancyGroupType" />
            <xsd:element name="AMS_RedundancyGroupRef"
               type="AMS_Ref" />
            <xsd:element name="AMS_LoadBalancingGroup"
               type="AMS_LoadBalancingGroupType" />
            <xsd:element name="AMS_LoadBalancingGroupRef"
               type="AMS_Ref" />
         </xsd:choice>
      </xsd:sequence>
   </xsd:complexType>

   <xsd:complexType name="AMS_ApplicationOfApplication_LinkType"
      mixed="false">
      <xsd:sequence minOccurs="0" maxOccurs="unbounded">
         <xsd:choice>
            <xsd:element name="AMS_Application"
               type="AMS_ApplicationType" />
            <xsd:element name="AMS_ApplicationRef" type="AMS_Ref" />
         </xsd:choice>
      </xsd:sequence>
   </xsd:complexType>

   <xsd:complexType name="AMS_ApplicationType">
      <xsd:complexContent>
         <xsd:extension base="CIM_ApplicationSystemType">
            <xsd:sequence>
               <xsd:element name="CIM_SystemComponent"
                  type="CIM_SystemComponent_Application_LinkType" minOccurs="0"
                  maxOccurs="unbounded" />
               <xsd:element name="AMS_ApplicationFeature"
                  type="AMS_Ref" minOccurs="1" maxOccurs="1" />
               <xsd:element name="AMS_ApplicationOfApplication"
                  type="AMS_ApplicationOfApplication_LinkType" minOccurs="0"
                  maxOccurs="unbounded" />
            </xsd:sequence>
            <xsd:attribute name="Status" type="AMS_RTSoftwareStatus"
               use="required" />
            <xsd:attribute name="Name" type="xsd:string"
               use="required" />
            <xsd:attribute name="OwnerId" type="xsd:string" />
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <xsd:simpleType name="AMS_BalancingStyle">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="LB_ROUND_ROBIN" />
         <xsd:enumeration value="LB_RANDOM" />
         <xsd:enumeration value="LB_IMPL_DEFINED" />
      </xsd:restriction>
   </xsd:simpleType>

   <xsd:simpleType name="AMS_StdState">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="ST_NONSTD"/>
         <xsd:enumeration value="ST_ENV"/>
      </xsd:restriction>
   </xsd:simpleType>
```

```
<xsd:complexType name="AMS_Property_StdStateType">
   <xsd:attribute name="Name" type="xsd:string" use="required"/>
   <xsd:attribute name="Value" type="xsd:string" use="required"/>
   <xsd:attribute name="InstanceID" type="xsd:string" use="required"/>
   <xsd:attribute name="StdName" type="AMS_StdState" use="required"/>
</xsd:complexType>

<xsd:complexType name="AMS_SpecificState_LinkType">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:element name="AMS_Property_StdState" type="AMS_Property_StdStateType"/>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="AMS_ExecutableSoftwareElementType"
   mixed="false">
   <xsd:complexContent>
      <xsd:extension base="CIM_ServiceType">
         <xsd:sequence>
            <xsd:element
               name="CIM_SoftwareElementServiceImplementation" type="AMS_Ref"
               minOccurs="1" maxOccurs="1" />
            <xsd:element name="CIM_RedundancyComponent"
               type="AMS_Ref" minOccurs="0" maxOccurs="1" />
            <xsd:element name="AMS_ESEDeployed" type="AMS_Ref"
               minOccurs="1" maxOccurs="1" />
            <xsd:element name="AMS_SpecificState" type="AMS_SpecificState_LinkType"
               minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
         <xsd:attribute name="CurrentState" type="AMS_State"
            use="required" />
         <xsd:attribute name="Status" type="AMS_RTSoftwareStatus"
            use="required" />
         <xsd:attribute name="RedState"
            type="AMS_RedundancyEltState" use="required" />
         <xsd:attribute name="Name" type="xsd:string"
            use="required" />
         <xsd:attribute name="OwnerId" type="xsd:string" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_LoadBalancingGroupType">
   <xsd:complexContent>
      <xsd:extension base="CIM_RedundancyGroupType">
         <xsd:sequence>
            <xsd:element name="AMS_LoadBalancingFeature"
               type="AMS_Ref" minOccurs="1" maxOccurs="1" />
            <xsd:element name="CIM_RedundancyComponent"
               type="AMS_Ref" minOccurs="1" maxOccurs="unbounded" />
         </xsd:sequence>
         <xsd:attribute name="Style" type="AMS_BalancingStyle"
            use="required" />
         <xsd:attribute name="Status" type="AMS_RTSoftwareStatus"
            use="required" />
         <xsd:attribute name="Name" type="xsd:string"
            use="required" />
         <xsd:attribute name="OwnerId" type="xsd:string" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

```
<xsd:simpleType name="AMS_RTSoftwareStatus">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="SW_STARTED" />
      <xsd:enumeration value="SW_STOPPED" />
      <xsd:enumeration value="SW_FAILED" />
   </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="AMS_RedundancyGroupType">
   <xsd:complexContent>
      <xsd:extension base="CIM_RedundancyGroupType">
         <xsd:sequence>
            <xsd:element name="CIM_RedundancyComponent"
               type="AMS_Ref" minOccurs="1" maxOccurs="unbounded" />
            <xsd:element name="AMS_RedundancyFeature"
               type="AMS_Ref" minOccurs="1" maxOccurs="1" />
         </xsd:sequence>
         <xsd:attribute name="Style" type="AMS_ReplicationStyle"
            use="required" />
         <xsd:attribute name="Status" type="AMS_RTSoftwareStatus"
            use="required" />
         <xsd:attribute name="Name" type="xsd:string"
            use="required" />
         <xsd:attribute name="OwnerId" type="xsd:string" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:simpleType name="AMS_ReplicationStyle">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="RG_COLD_PASSIVE" />
      <xsd:enumeration value="RG_WARM_PASSIVE" />
      <xsd:enumeration value="RG_ACTIVE" />
      <xsd:enumeration value="RG_ACTIVE_WITH_VOTING" />
      <xsd:enumeration value="RG_STATELESS" />
      <xsd:enumeration value="RG_IMPL_DEFINED" />
   </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="CIM_SystemComponent_SoftwareSystem_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="AMS_Application"
            type="AMS_ApplicationType" />
         <xsd:element name="AMS_ApplicationRef" type="AMS_Ref" />
         <xsd:element name="AMS_ExecutableSoftwareElement"
            type="AMS_ExecutableSoftwareElementType" />
         <xsd:element name="AMS_ExecutableSoftwareElementRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="AMS_SystemOfSystem_LinkType">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="AMS_SoftwareSystem"
            type="AMS_SoftwareSystemType" />
```

```
            <xsd:element name="AMS_SoftwareSystemRef"
               type="AMS_Ref" />
         </xsd:choice>
      </xsd:sequence>
   </xsd:complexType>

   <xsd:complexType name="AMS_SoftwareSystemType">
      <xsd:complexContent>
         <xsd:extension base="CIM_ApplicationSystemType">
            <xsd:sequence>
               <xsd:element name="CIM_SystemComponent"
                  type="CIM_SystemComponent_SoftwareSystem_LinkType" minOccurs="0"
                  maxOccurs="unbounded" />
               <xsd:element name="AMS_SystemOfSystem"
                  type="AMS_SystemOfSystem_LinkType" minOccurs="0"
                  maxOccurs="unbounded" />
               <xsd:element name="AMS_SystemFeature" type="AMS_Ref"
                  minOccurs="1" maxOccurs="1" />
            </xsd:sequence>
            <xsd:attribute name="Status" type="AMS_RTSoftwareStatus"
               use="required" />
            <xsd:attribute name="Name" type="xsd:string"
               use="required" />
            <xsd:attribute name="OwnerId" type="xsd:string" />
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <!-- structure definition !-->
   <xsd:element name="AMS_Application" type="AMS_ApplicationPackage" />
   <xsd:complexType name="AMS_ApplicationPackage">
      <xsd:all minOccurs="0">
         <xsd:element name="AMS_Application"
            type="AMS_ApplicationType" minOccurs="0" />
         <xsd:element name="AMS_ExecutableSoftwareElement"
            type="AMS_ExecutableSoftwareElementType" minOccurs="0" />
         <xsd:element name="AMS_LoadBalancingGroup"
            type="AMS_LoadBalancingGroupType" minOccurs="0" />
         <xsd:element name="AMS_RedundancyGroup"
            type="AMS_RedundancyGroupType" minOccurs="0" />
         <xsd:element name="AMS_SoftwareSystem"
            type="AMS_SoftwareSystemType" minOccurs="0" />
      </xsd:all>
   </xsd:complexType>
</xsd:schema>
```

9.6.4 Application Deployment

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:amsm="http://www.omg.org/amsm"
   targetNamespace="http://www.omg.org/amsm"
   xmlns="http://www.omg.org/amsm">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         file AMS_ApplicationDeployment.xsd
         XML Schema Definition for namespace AMS Management
         Copyright (c) 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
               Themis Computer and ProgenySystems Corporation
      </xsd:documentation>
   </xsd:annotation>
```

```
<xsd:include schemaLocation="AMS_Util.xsd" />
<xsd:include schemaLocation="AMS_CIM.xsd" />
<xsd:include schemaLocation="AMS_Application.xsd" />
<xsd:include schemaLocation="AMS_ApplicationSpecification.xsd" />
<xsd:include
   schemaLocation="AMS_ApplicationDeploymentSpecification.xsd" />
<xsd:include schemaLocation="AMS_LogicalHardware.xsd" />
<!-- types definition !-->
<xsd:complexType name="AMS_DeploymentLinks_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="AMS_DeploymentLink"
            type="AMS_DeploymentLinkType" />
         <xsd:element name="AMS_DeploymentLinkRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="AMS_DeploymentConfigurationType"
   mixed="false">
   <xsd:complexContent>
      <xsd:extension base="CIM_LogicalElementType">
         <xsd:sequence>
            <xsd:element name="AMS_DeploymentLinks"
               type="AMS_DeploymentLinks_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
            <xsd:element name="AMS_DeploymentSpecAssoc"
               type="AMS_Ref" minOccurs="1" maxOccurs="1" />
         </xsd:sequence>
         <xsd:attribute name="Name" type="xsd:string"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_DeploymentLinkType">
   <xsd:complexContent>
      <xsd:extension base="CIM_LogicalElementType">
         <xsd:sequence>
            <xsd:element name="AMS_HostUsed" type="AMS_Ref"
               minOccurs="1" maxOccurs="1" />
            <xsd:element name="AMS_ESEDeployed" type="AMS_Ref"
               minOccurs="1" maxOccurs="1" />
         </xsd:sequence>
         <xsd:attribute name="LinkID" type="xsd:string" />
         <xsd:attribute name="OwnerId" type="xsd:string" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<!-- structure definition !-->
<xsd:element name="AMS_ApplicationDeployment"
   type="AMS_ApplicationDeploymentPackage" />
<xsd:complexType name="AMS_ApplicationDeploymentPackage"
   mixed="false">
   <xsd:all minOccurs="0">
      <xsd:element name="AMS_DeploymentConfiguration"
         type="AMS_DeploymentConfigurationType" minOccurs="0" />
```

```
            <xsd:element name="AMS_DeploymentLink"
               type="AMS_DeploymentLinkType" minOccurs="0" />
         </xsd:all>
      </xsd:complexType>
   </xsd:schema>
```

9.6.5 Application Deployment Specification

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:amsm="http://www.omg.org/amsm"
   targetNamespace="http://www.omg.org/amsm"
   xmlns="http://www.omg.org/amsm">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         file AMS_ApplicationDeploymentSpecification.xsd
         XML Schema Definition for namespace AMS Management
         Copyright (c) 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
               Themis Computer and ProgenySystems Corporation
      </xsd:documentation>
   </xsd:annotation>
   <xsd:include schemaLocation="AMS_Util.xsd" />
   <xsd:include schemaLocation="AMS_CIM.xsd" />
   <xsd:include schemaLocation="AMS_LogicalHardware.xsd" />
   <xsd:include schemaLocation="AMS_ApplicationSpecification.xsd" />
   <!-- types definition !-->
   <xsd:complexType name="AMS_ActionOnLink_LinkType">
      <xsd:sequence minOccurs="0" maxOccurs="1">
         <xsd:choice>
            <xsd:element name="CIM_Action" type="CIM_ActionType" />
            <xsd:element name="CIM_ExecuteProgram"
               type="CIM_ExecuteProgramType" />
            <xsd:element name="AMS_ExecuteProgram"
               type="AMS_ExecuteProgramType" />
            <xsd:element name="CIM_FileAction"
               type="CIM_FileActionType" />
            <xsd:element name="CIM_CopyFileAction"
               type="CIM_CopyFileActionType" />
            <xsd:element name="AMS_CCMDeploy"
               type="AMS_CCMDeployType" />
            <xsd:element name="AMS_CCMStart"
               type="AMS_CCMStartType" />
            <xsd:element name="AMS_CCMStop"
               type="AMS_CCMStopType" />
            <xsd:element name="AMS_J2EEDeploy"
               type="AMS_J2EEDeployType" />
            <xsd:element name="AMS_J2EEStart"
               type="AMS_J2EEStartType" />
            <xsd:element name="AMS_J2EEStop"
               type="AMS_J2EEStopType" />
            <xsd:element name="CIM_ActionRef" type="AMS_Ref" />
         </xsd:choice>
      </xsd:sequence>
      <xsd:attribute name="Case" type="AMS_ActionCheckCase"
         use="required" />
   </xsd:complexType>

   <xsd:complexType name="AMS_DeploymentLinkSpecType">
      <xsd:complexContent>
         <xsd:extension base="CIM_LogicalElementType">
            <xsd:sequence>
```

```
            <xsd:element name="AMS_ConfSpecDLS" type="AMS_Ref"
               minOccurs="0" maxOccurs="1" />
            <xsd:element name="AMS_SEDeployed" type="AMS_Ref"
               minOccurs="0" maxOccurs="1" />
            <xsd:element name="AMS_DeploymentLinkDependency"
               type="AMS_Ref" minOccurs="0" maxOccurs="1" />
            <xsd:element name="AMS_ActionOnLink"
               type="AMS_ActionOnLink_LinkType" minOccurs="0" maxOccurs="1" />
            <xsd:element name="AMS_HostUsed" type="AMS_Ref"
               minOccurs="0" maxOccurs="unbounded" />
            <xsd:element name="AMS_OSUsed" type="AMS_Ref"
               minOccurs="0" maxOccurs="unbounded" />
         </xsd:sequence>
         <xsd:attribute name="LinkID" type="xsd:string" />
         <xsd:attribute name="OwnerId" type="xsd:string" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_DeploymentSpecLinks_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="AMS_DeploymentLinkSpec"
            type="AMS_DeploymentLinkSpecType" />
         <xsd:element name="AMS_DeploymentLinkSpecRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="AMS_DeploymentSpecType">
   <xsd:complexContent>
      <xsd:extension base="CIM_LogicalElementType">
         <xsd:sequence>
            <xsd:element name="AMS_DeploymentSpecAssoc"
               type="AMS_Ref" minOccurs="0" maxOccurs="unbounded" />
            <xsd:element name="AMS_DeploymentSpecLinks"
               type="AMS_DeploymentSpecLinks_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
         </xsd:sequence>
         <xsd:attribute name="Name" type="xsd:string"
            use="required" />
         <xsd:attribute name="OwnerId" type="xsd:string" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<!-- structure definition !-->
<xsd:element name="AMS_ApplicationDeploymentSpecification"
   type="AMS_ApplicationDeploymentSpecificationPackage" />
<xsd:complexType
   name="AMS_ApplicationDeploymentSpecificationPackage">
   <xsd:all minOccurs="0">
      <xsd:element name="AMS_DeploymentLinkSpec"
         type="AMS_DeploymentLinkSpecType" minOccurs="0" />
      <xsd:element name="AMS_DeploymentSpec"
         type="AMS_DeploymentSpecType" minOccurs="0" />
   </xsd:all>
</xsd:complexType>
```

```
</xsd:schema>
```

9.6.6 Application Specification

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:amsm="http://www.omg.org/amsm"
   targetNamespace="http://www.omg.org/amsm"
   xmlns="http://www.omg.org/amsm">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         file AMS_ApplicationSpecification.xsd
         XML Schema Definition for namespace AMS Management
         Copyright (c) 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
               Themis Computer and ProgenySystems Corporation
      </xsd:documentation>
   </xsd:annotation>
   <xsd:include schemaLocation="AMS_Util.xsd" />
   <xsd:include schemaLocation="AMS_CIM.xsd" />
   <xsd:include schemaLocation="AMS_SupportedApplicationModel.xsd" />
   <!-- types definition !-->
   <xsd:simpleType name="AMS_ActionCheckCase">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="CASE_PRE_DEPLOY" />
         <xsd:enumeration value="CASE_DEPLOY" />
         <xsd:enumeration value="CASE_POST_DEPLOY" />
         <xsd:enumeration value="CASE_PRE_START" />
         <xsd:enumeration value="CASE_START" />
         <xsd:enumeration value="CASE_POST_START" />
         <xsd:enumeration value="CASE_PRE_SHUTDOWN" />
         <xsd:enumeration value="CASE_SHUTDOWN" />
         <xsd:enumeration value="CASE_POST_SHUTDOWN" />
         <xsd:enumeration value="CASE_ALTERNATE_SHUTDOWN" />
      </xsd:restriction>
   </xsd:simpleType>

   <xsd:complexType name="AMS_ApplicationModelCheckType"
      mixed="false">
      <xsd:complexContent>
         <xsd:extension base="CIM_CheckType">
            <xsd:sequence>
               <xsd:element name="AMS_SAMCheck" type="AMS_Ref"
                  minOccurs="1" maxOccurs="1" />
            </xsd:sequence>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <xsd:complexType name="AMS_SecurityCheckType"
      mixed="false">
      <xsd:complexContent>
         <xsd:extension base="CIM_CheckType"></xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <xsd:complexType name="AMS_SoftwareElementCheck_LinkType"
      mixed="false">
      <xsd:sequence minOccurs="0" maxOccurs="unbounded">
         <xsd:choice>
            <xsd:element name="CIM_Check" type="CIM_CheckType" />
            <xsd:element name="AMS_ApplicationModelCheck"
```

```
            type="AMS_ApplicationModelCheckType" />
         <xsd:element name="AMS_SecurityCheck"
            type="AMS_SecurityCheckType" />
         <xsd:element name="CIM_OSVersionCheck"
            type="CIM_OSVersionCheckType" />
         <xsd:element name="CIM_ArchitectureCheck"
            type="CIM_ArchitectureCheckType" />
         <xsd:element name="CIM_CheckRef" type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
   <xsd:attribute name="Case" type="AMS_ActionCheckCase"
      use="required" />
</xsd:complexType>

<xsd:complexType name="AMS_SoftwareElementAction_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="CIM_Action" type="CIM_ActionType" />
         <xsd:element name="AMS_MechanizedAction"
            type="AMS_MechanizedActionType" />
         <xsd:element name="CIM_ExecuteProgram"
            type="CIM_ExecuteProgramType" />
         <xsd:element name="AMS_ExecuteProgram"
            type="AMS_ExecuteProgramType" />
         <xsd:element name="CIM_FileAction"
            type="CIM_FileActionType" />
         <xsd:element name="CIM_CopyFileAction"
            type="CIM_CopyFileActionType" />
         <xsd:element name="AMS_CCMDeploy"
            type="AMS_CCMDeployType" />
         <xsd:element name="AMS_CCMStart"
            type="AMS_CCMStartType" />
         <xsd:element name="AMS_CCMStop"
            type="AMS_CCMStopType" />
         <xsd:element name="AMS_J2EEDeploy"
            type="AMS_J2EEDeployType" />
         <xsd:element name="AMS_J2EEStart"
            type="AMS_J2EEStartType" />
         <xsd:element name="AMS_J2EEStop"
            type="AMS_J2EEStopType" />
         <xsd:element name="CIM_ActionRef" type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
   <xsd:attribute name="Case" type="AMS_ActionCheckCase"
      use="required" />
</xsd:complexType>

<xsd:complexType name="AMS_SEStartDependency_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:element name="Ref" type="AMS_Ref" />
      <xsd:choice minOccurs="1" maxOccurs="1">
         <xsd:element name="AMS_SEStartTimeDependency"
            type="AMS_SEStartTimeDependencyType" />
         <xsd:element name="AMS_SEStartCPUDependency"
            type="AMS_SEStartCPUDependencyType" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>
```

```
<xsd:complexType name="AMS_SEShutdownDependency_LinkType"
   mixed="false">
   <xsd:sequence>
      <xsd:element name="Ref" type="AMS_Ref" minOccurs="0"
         maxOccurs="unbounded" />
      <xsd:element name="AMS_SEShutdownDependency"
         type="AMS_SEShutdownDependencyType" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="AMS_ESESpecType">
   <xsd:complexContent>
      <xsd:extension base="CIM_SoftwareElementType">
         <xsd:sequence>
            <xsd:element name="AMS_SoftwareElementCheck"
               type="AMS_SoftwareElementCheck_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
            <xsd:element name="AMS_SoftwareElementAction"
               type="AMS_SoftwareElementAction_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
            <xsd:element name="AMS_SEStartDependency"
               type="AMS_SEStartDependency_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
            <xsd:element name="AMS_SEShutdownDependency"
               type="AMS_SEShutdownDependency_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
         </xsd:sequence>
         <xsd:attribute name="ModelType" type="AMS_ModelType"
            use="required" />
         <xsd:attribute name="RedInitState"
            type="AMS_RedundancyEltState" use="required" />
         <xsd:attribute name="Name" type="xsd:string"
            use="required" />
         <xsd:attribute name="SoftwareElementID"
            type="xsd:string" />
         <xsd:attribute name="OwnerId" type="xsd:string" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_Property_StdMechanismType">
   <xsd:attribute name="Name" type="xsd:string"
            use="required" />
   <xsd:attribute name="Value" type="xsd:string" use="required"/>
   <xsd:attribute name="instanceID" type="xsd:string" use="required"/>
   <xsd:attribute name="StdMechanism" type="AMS_StdMechanism" use="required"/>
</xsd:complexType>

<xsd:complexType name="AMS_ActionMechanism_LinkType">
   <xsd:sequence minOccurs="1" maxOccurs="1">
      <xsd:choice>
         <xsd:element name="AMS_Property_StdMechanism"
                     type="AMS_Property_StdMechanismType"/>
         <xsd:element name="AMS_Property_StdMechanismRef" type="AMS_Ref"/>
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="AMS_MechanizedActionType">
```

```
   <xsd:complexContent>
      <xsd:extension base="CIM_ActionType">
         <xsd:sequence>
            <xsd:element name="AMS_ActionMechanism"
                     type="AMS_ActionMechanism_LinkType"
                     minOccurs="1" maxOccurs="1"/>
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_ExecuteProgramType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ExecuteProgramType">
         <xsd:attribute name="Environment" type="xsd:string"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_CCMDeployType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ActionType">
         <xsd:attribute name="ComponentPackageDescriptor" type="xsd:string"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_CCMStartType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ActionType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_CCMStopType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ActionType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_J2EEDeployType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ActionType">
         <xsd:attribute name="EntrepriseARchive" type="xsd:string"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_J2EEStartType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ActionType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_J2EEStopType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ActionType"></xsd:extension>
   </xsd:complexContent>
```

```
</xsd:complexType>

<xsd:simpleType name="AMS_RedundancyEltState">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="REDSTATE_NORG" />
      <xsd:enumeration value="REDSTATE_PRIMARY" />
      <xsd:enumeration value="REDSTATE_PASSIVE" />
   </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="AMS_SEShutdownDependencyType"
   mixed="false">
   <xsd:complexContent>
      <xsd:extension base="CIM_DependencyType">
         <xsd:attribute name="TimeSinceShutdown" type="xsd:int"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_SEStartCPUDependencyType"
   mixed="false">
   <xsd:complexContent>
      <xsd:extension base="AMS_SEStartDependencyType">
         <xsd:attribute name="CPUload" type="xsd:int"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_SEStartDependencyType">
   <xsd:complexContent>
      <xsd:extension base="CIM_DependencyType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_SEStartTimeDependencyType"
   mixed="false">
   <xsd:complexContent>
      <xsd:extension base="AMS_SEStartDependencyType">
         <xsd:attribute name="TimeSinceStartup" type="xsd:int"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_FeatureOfFeature_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="AMS_SoftwareFeatureSpec"
            type="AMS_SoftwareFeatureSpecType" />
         <xsd:element name="AMS_SoftwareFeatureSpecRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CIM_SoftwareFeatureSoftwareElement_LinkType"
   mixed="false">
```

```
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="AMS_ESESpec" type="AMS_ESESpecType" />
         <xsd:element name="AMS_ESESpecRef" type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="AMS_SoftwareFeatureCheck_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="CIM_Check" type="CIM_CheckType" />
         <xsd:element name="AMS_ApplicationModelCheck"
            type="AMS_ApplicationModelCheckType" />
         <xsd:element name="AMS_SecurityCheck"
            type="AMS_SecurityCheckType" />
         <xsd:element name="CIM_OSVersionCheck"
            type="CIM_OSVersionCheckType" />
         <xsd:element name="CIM_ArchitectureCheck"
            type="CIM_ArchitectureCheckType" />
         <xsd:element name="CIM_CheckRef" type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
   <xsd:attribute name="Case" type="AMS_ActionCheckCase"
      use="required" />
</xsd:complexType>

<xsd:complexType name="AMS_SoftwareFeatureAction_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="CIM_Action" type="CIM_ActionType" />
         <xsd:element name="AMS_MechanizedAction"
            type="AMS_MechanizedActionType" />
         <xsd:element name="CIM_ExecuteProgram"
            type="CIM_ExecuteProgramType" />
         <xsd:element name="AMS_ExecuteProgram"
            type="AMS_ExecuteProgramType" />
         <xsd:element name="CIM_FileAction"
            type="CIM_FileActionType" />
         <xsd:element name="CIM_CopyFileAction"
            type="CIM_CopyFileActionType" />
         <xsd:element name="AMS_CCMDeploy"
            type="AMS_CCMDeployType" />
         <xsd:element name="AMS_CCMStart"
            type="AMS_CCMStartType" />
         <xsd:element name="AMS_CCMStop"
            type="AMS_CCMStopType" />
         <xsd:element name="AMS_J2EEDeploy"
            type="AMS_J2EEDeployType" />
         <xsd:element name="AMS_J2EEStart"
            type="AMS_J2EEStartType" />
         <xsd:element name="AMS_J2EEStop"
            type="AMS_J2EEStopType" />
         <xsd:element name="CIM_ActionRef" type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
   <xsd:attribute name="Case" type="AMS_ActionCheckCase"
      use="required" />
```

```
</xsd:complexType>

<xsd:complexType name="AMS_SoftwareFeatureSpecType">
   <xsd:complexContent>
      <xsd:extension base="CIM_SoftwareFeatureType">
         <xsd:sequence>
            <xsd:element name="AMS_FeatureOfFeature"
               type="AMS_FeatureOfFeature_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
            <xsd:element
               name="CIM_SoftwareFeatureSoftwareElement"
               type="CIM_SoftwareFeatureSoftwareElement_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
            <xsd:element name="AMS_SoftwareFeatureCheck"
               type="AMS_SoftwareFeatureCheck_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
            <xsd:element name="AMS_SoftwareFeatureAction"
               type="AMS_SoftwareFeatureAction_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
         </xsd:sequence>
         <xsd:attribute name="TypeOfFeature"
            type="AMS_TypeOfFeature" use="required" />
         <xsd:attribute name="Name" type="xsd:string"
            use="required" />
         <xsd:attribute name="OwnerId" type="xsd:string" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:simpleType name="AMS_TypeOfFeature">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="SYSTEM" />
      <xsd:enumeration value="APPLICATION" />
      <xsd:enumeration value="REDUNDANCY_GROUP" />
      <xsd:enumeration value="LOADBALANCING_GROUP" />
   </xsd:restriction>
</xsd:simpleType>

<!-- structure definition !-->
<xsd:element name="AMS_ApplicationSpecification"
   type="AMS_ApplicationSpecificationPackage" />
<xsd:complexType name="AMS_ApplicationSpecificationPackage"
   mixed="false">
   <xsd:all minOccurs="0">
      <xsd:element name="AMS_ApplicationModelCheck"
         type="AMS_ApplicationModelCheckType" minOccurs="0" />
      <xsd:element name="AMS_SecurityCheck"
            type="AMS_SecurityCheckType" minOccurs="0" />
      <xsd:element name="CIM_OSVersionCheck"
         type="CIM_OSVersionCheckType" minOccurs="0" />
      <xsd:element name="CIM_ArchitectureCheck"
         type="CIM_ArchitectureCheckType" minOccurs="0" />

      <xsd:element name="AMS_MechanizedAction"
         type="AMS_MechanizedActionType" minOccurs="0" />
      <xsd:element name="AMS_ExecuteProgram"
         type="AMS_ExecuteProgramType" minOccurs="0" />
      <xsd:element name="AMS_CCMDeploy"
         type="AMS_CCMDeployType" minOccurs="0" />
      <xsd:element name="AMS_CCMStart"
```

```
                  type="AMS_CCMStartType" minOccurs="0" />
               <xsd:element name="AMS_CCMStop"
                  type="AMS_CCMStopType" minOccurs="0" />
               <xsd:element name="AMS_J2EEDeploy"
                  type="AMS_J2EEDeployType" minOccurs="0" />
               <xsd:element name="AMS_J2EEStart"
                  type="AMS_J2EEStartType" minOccurs="0" />
               <xsd:element name="AMS_J2EEStop"
                  type="AMS_J2EEStopType" minOccurs="0" />
               <xsd:element name="CIM_ExecuteProgram"
                  type="CIM_ExecuteProgramType" minOccurs="0" />
               <xsd:element name="CIM_FileAction" type="CIM_FileActionType"
                  minOccurs="0" />
               <xsd:element name="CIM_CopyFileAction"
                  type="CIM_CopyFileActionType" minOccurs="0" />

               <xsd:element name="AMS_ESESpec" type="AMS_ESESpecType"
                  minOccurs="0" />

               <xsd:element name="AMS_SoftwareFeatureSpec"
                  type="AMS_SoftwareFeatureSpecType" minOccurs="0" />
            </xsd:all>
         </xsd:complexType>
      </xsd:schema>
```

9.6.7 CIM

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:amsm="http://www.omg.org/amsm"
   targetNamespace="http://www.omg.org/amsm"
   xmlns="http://www.omg.org/amsm">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         file AMS_CIM.xsd
         XML Schema Definition for namespace AMS Management
         Copyright (c) 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
               Themis Computer and ProgenySystems Corporation
      </xsd:documentation>
   </xsd:annotation>
   <xsd:include schemaLocation="AMS_Util.xsd" />
   <xsd:include schemaLocation="AMS_LogicalHardware.xsd" />
   <!-- types definition !-->
   <xsd:complexType name="CIM_ActionSequence_LinkType">
      <xsd:sequence minOccurs="0" maxOccurs="unbounded">
         <xsd:choice>
            <xsd:element name="CIM_Action" type="CIM_ActionType" />
            <xsd:element name="CIM_ActionRef" type="AMS_Ref" />
         </xsd:choice>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="CIM_ActionType">
      <xsd:sequence>
         <xsd:element name="CIM_ActionSequence"
            type="CIM_ActionSequence_LinkType" minOccurs="0"
            maxOccurs="unbounded" />
      </xsd:sequence>
      <xsd:attribute name="ActionID" type="xsd:string" />
      <xsd:attribute name="TargetOperatingSystem" type="xsd:int"
         use="required" />
      <xsd:attribute name="OwnerId" type="xsd:string" />
```

```
</xsd:complexType>
<xsd:complexType name="CIM_AdminDomainType">
   <xsd:attribute name="Name" type="xsd:string" use="required" />
   <xsd:attribute name="OwnerId" type="xsd:string" />
</xsd:complexType>
<xsd:complexType name="CIM_ApplicationSystemType">
   <xsd:complexContent>
      <xsd:extension base="CIM_SystemType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_ArchitectureCheckType">
   <xsd:complexContent>
      <xsd:extension base="CIM_CheckType">
         <xsd:attribute name="ArchitectureType" type="xsd:int"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_CheckType">
   <xsd:attribute name="CheckID" type="xsd:string" />
   <xsd:attribute name="OwnerId" type="xsd:string" />
</xsd:complexType>
<xsd:complexType name="CIM_ComputerSystemType">
   <xsd:complexContent>
      <xsd:extension base="CIM_SystemType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_MemberOfCollection_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="CIM_ProtocolEndPoint"
            type="CIM_ProtocolEndPointType" />
         <xsd:element name="CIM_LANEndPoint"
            type="CIM_LANEndPointType" />
         <xsd:element name="CIM_IPProtocolEndPoint"
            type="CIM_IPProtocolEndPointType" />
         <xsd:element name="AMS_LANEndPoint"
            type="AMS_LANEndPointType" />
         <xsd:element name="CIM_ProtocolEndPointRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CIM_ConnectivityCollectionType"
   mixed="false">
   <xsd:sequence>
      <xsd:element name="CIM_MemberOfCollection"
         type="CIM_MemberOfCollection_LinkType" minOccurs="0"
         maxOccurs="unbounded" />
   </xsd:sequence>
   <xsd:attribute name="InstanceID" type="xsd:integer" />
   <xsd:attribute name="OwnerId" type="xsd:string" />
</xsd:complexType>
<xsd:complexType name="CIM_ToDirectoryAction_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="1" maxOccurs="1">
      <xsd:choice>
         <xsd:element name="CIM_DirectorySpecification"
            type="CIM_DirectorySpecificationType" />
```

```
         <xsd:element name="CIM_DirectorySpecificationRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CIM_FromDirectorySpecification_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="1" maxOccurs="1">
      <xsd:choice>
         <xsd:element name="CIM_DirectorySpecification"
            type="CIM_DirectorySpecificationType" />
         <xsd:element name="CIM_DirectorySpecificationRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CIM_CopyFileActionType">
   <xsd:complexContent>
      <xsd:extension base="CIM_FileActionType">
         <xsd:sequence>
            <xsd:element name="CIM_ToDirectoryAction"
               type="CIM_ToDirectoryAction_LinkType" minOccurs="1" maxOccurs="1" />
            <xsd:element name="CIM_FromDirectorySpecification"
               type="CIM_FromDirectorySpecification_LinkType" minOccurs="1"
               maxOccurs="1" />
         </xsd:sequence>
         <xsd:attribute name="Source" type="xsd:string"
            use="required" />
         <xsd:attribute name="Destination" type="xsd:string"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_DependencyType">
</xsd:complexType>
<xsd:complexType name="CIM_DirectorySpecificationType"
   mixed="false">
   <xsd:complexContent>
      <xsd:extension base="CIM_CheckType">
         <xsd:attribute name="DirectoryPath" type="xsd:string"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_DisplayType">
   <xsd:complexContent>
      <xsd:extension base="CIM_LogicalDeviceType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_EnabledLogicalElementType"
   mixed="false">
   <xsd:complexContent>
      <xsd:extension base="CIM_LogicalElementType">
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_EthernetPortType">
   <xsd:complexContent>
      <xsd:extension base="CIM_NetworkPortType"></xsd:extension>
   </xsd:complexContent>
```

```
</xsd:complexType>
<xsd:complexType name="CIM_EthernetPortStatisticsType"
   mixed="false">
   <xsd:attribute name="InstanceID" type="xsd:integer" />
   <xsd:attribute name="BytesTransmitted" type="xsd:integer"
      use="required" />
   <xsd:attribute name="BytesReceived" type="xsd:integer"
      use="required" />
   <xsd:attribute name="PacketsTransmitted" type="xsd:integer"
      use="required" />
   <xsd:attribute name="PacketsReceived" type="xsd:integer"
      use="required" />
</xsd:complexType>
<xsd:complexType name="CIM_ExecuteProgramType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ActionType">
         <xsd:attribute name="CommandLine" type="xsd:string"
            use="required" />
         <xsd:attribute name="ProgramPath" type="xsd:string"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_FileActionType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ActionType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_IPProtocolEndPointType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ProtocolEndPointType">
         <xsd:attribute name="IPv4Address" type="xsd:string" />
         <xsd:attribute name="IPv6Address" type="xsd:string" />
         <xsd:attribute name="SubnetMask" type="xsd:string" />
         <xsd:attribute name="PrefixLength" type="xsd:short" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_IndicationType">
   <xsd:attribute name="IndicationTime" type="xsd:string"
      use="required" />
</xsd:complexType>
<xsd:complexType name="CIM_ElementStatisticalData_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="CIM_EthernetPortStatistics"
            type="CIM_EthernetPortStatisticsType" />
         <xsd:element name="CIM_EthernetPortStatisticsRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CIM_LANEndPointType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ProtocolEndPointType">
         <xsd:sequence>
            <xsd:element name="CIM_ElementStatisticalData"
               type="CIM_ElementStatisticalData_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
```

```
            <xsd:element name="CIM_BindsToLANEndPoint"
               type="AMS_Ref" minOccurs="0" maxOccurs="unbounded" />
         </xsd:sequence>
         <xsd:attribute name="LANID" type="xsd:string" />
         <xsd:attribute name="MACAddress" type="xsd:string"
            use="required" />
         <xsd:attribute name="AliasAddresses" type="xsd:string"
            use="required" />
         <xsd:attribute name="GroupAddresses" type="xsd:string"
            use="required" />
         <xsd:attribute name="MaxDataSize" type="xsd:long"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_LocationType">
   <xsd:sequence>
      <xsd:element name="CIM_PhysicalElementLocation"
         type="AMS_Ref" minOccurs="0" maxOccurs="unbounded" />
      <xsd:element name="CIM_ElementLocation" type="AMS_Ref"
         minOccurs="0" maxOccurs="unbounded" />
   </xsd:sequence>
   <xsd:attribute name="Name" type="xsd:string" use="required" />
</xsd:complexType>
<xsd:complexType name="CIM_LogType">
   <xsd:complexContent>
      <xsd:extension base="CIM_EnabledLogicalElementType">
         <xsd:attribute name="MaxNumberOfRecords"
            type="xsd:integer" use="required" />
         <xsd:attribute name="CurrentNumberOfRecord"
            type="xsd:integer" use="required" />
         <xsd:attribute name="Name" type="xsd:string"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_LogicalDeviceType">
   <xsd:sequence>
      <xsd:element name="CIM_Realizes" type="AMS_Ref"
         minOccurs="0" maxOccurs="unbounded" />
   </xsd:sequence>
   <xsd:attribute name="DeviceID" type="xsd:string" />
   <xsd:attribute name="OwnerId" type="xsd:string" />
</xsd:complexType>
<xsd:complexType name="CIM_LogicalDiskType">
   <xsd:complexContent>
      <xsd:extension base="CIM_StorageExtentType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_LogicalElementType">
</xsd:complexType>
<xsd:complexType name="CIM_LogicalPortType">
   <xsd:complexContent>
      <xsd:extension base="CIM_LogicalDeviceType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_ManagedElementType">
</xsd:complexType>
<xsd:complexType name="CIM_MemoryType">
   <xsd:complexContent>
```

```
      <xsd:extension base="CIM_LogicalDeviceType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_HostedCollection_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="CIM_ConnectivityCollection"
            type="CIM_ConnectivityCollectionType" />
         <xsd:element name="CIM_ConnectivityCollectionRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="AMS_SubnetComponent_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="CIM_Network" type="CIM_NetworkType" />
         <xsd:element name="CIM_NetworkRef" type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CIM_NetworkType">
   <xsd:complexContent>
      <xsd:extension base="CIM_AdminDomainType">
         <xsd:sequence>
            <xsd:element name="CIM_HostedCollection"
               type="CIM_HostedCollection_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
            <xsd:element name="AMS_SubnetComponent"
               type="AMS_SubnetComponent_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_NetworkPortType">
   <xsd:complexContent>
      <xsd:extension base="CIM_LogicalPortType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_NextHopIPRouteType">
   <xsd:complexContent>
      <xsd:extension base="CIM_NextHopRouteType">
         <xsd:attribute name="RouteDerivation" type="xsd:int"
            use="required" />
         <xsd:attribute name="DestinationMask" type="xsd:string"
            use="required" />
         <xsd:attribute name="PrefixLength" type="xsd:short"
            use="required" />
         <xsd:attribute name="AddressType" type="xsd:int"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_NextHopRouteType">
   <xsd:sequence>
      <xsd:element name="CIM_AssociatedNextHop" type="AMS_Ref"
         minOccurs="0" maxOccurs="1" />
```

```
      <xsd:element name="CIM_RouteUsesEndpoint" type="AMS_Ref"
         minOccurs="0" maxOccurs="1" />
   </xsd:sequence>
   <xsd:attribute name="InstanceID" type="xsd:string" />
   <xsd:attribute name="DestinationAddress" type="xsd:string"
      use="required" />
   <xsd:attribute name="AdminDistance" type="xsd:int"
      use="required" />
   <xsd:attribute name="RouteMetric" type="xsd:int" use="required" />
   <xsd:attribute name="IsStatic" type="xsd:boolean"
      use="required" />
   <xsd:attribute name="TypeOfRoute" type="xsd:int" use="required" />
</xsd:complexType>
<xsd:complexType name="CIM_OSVersionCheckType">
   <xsd:complexContent>
      <xsd:extension base="CIM_CheckType">
         <xsd:attribute name="MaximumVersion" type="xsd:string"
            use="required" />
         <xsd:attribute name="MinimumVersion" type="xsd:string"
            use="required" />
         <xsd:attribute name="TargetOperatingSystem"
            type="xsd:int" use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_OperatingSystemType">
</xsd:complexType>
<xsd:complexType name="CIM_PhysicalElementType">
   <xsd:sequence>
      <xsd:element name="CIM_Realizes" type="AMS_Ref"
         minOccurs="0" maxOccurs="unbounded" />
      <xsd:element name="CIM_PhysicalElementLocation"
         type="AMS_Ref" minOccurs="0" maxOccurs="1" />
   </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CIM_PowerSupplyType">
   <xsd:complexContent>
      <xsd:extension base="CIM_LogicalDeviceType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_ProcessIndicationType">
   <xsd:complexContent>
      <xsd:extension base="CIM_IndicationType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_ProcessorType">
   <xsd:complexContent>
      <xsd:extension base="CIM_LogicalDeviceType">
         <xsd:attribute name="LoadPercentage" type="xsd:int"
            use="required" />
         <xsd:attribute name="CPUStatus" type="xsd:int"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_ProtocolEndPointType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ServiceAccessPointType">
         <xsd:sequence>
            <xsd:element name="CIM_EndpointIdentity"
```

```
               type="AMS_Ref" minOccurs="0" maxOccurs="unbounded" />
            <xsd:element name="CIM_RouteUsesEndpoint"
               type="AMS_Ref" minOccurs="0" maxOccurs="unbounded" />
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_RecordForLogType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ManagedElementType">
         <xsd:attribute name="RecordFormat" type="xsd:string"
            use="required" />
         <xsd:attribute name="RecordData" type="xsd:string"
            use="required" />
         <xsd:attribute name="Locale" type="xsd:string"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_RedundancyGroupType">
   <xsd:complexContent>
      <xsd:extension base="CIM_LogicalElementType">
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_RemoteServiceAccessPointType"
   mixed="false">
   <xsd:complexContent>
      <xsd:extension base="CIM_ServiceAccessPointType">
         <xsd:sequence>
            <xsd:element name="CIM_AssociatedNextHop"
               type="AMS_Ref" minOccurs="0" maxOccurs="unbounded" />
         </xsd:sequence>
         <xsd:attribute name="AccessInfo" type="xsd:string"
            use="required" />
         <xsd:attribute name="InfoFormat" type="xsd:int"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_SensorType">
   <xsd:complexContent>
      <xsd:extension base="CIM_LogicalDeviceType"></xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_ThreadType">
   <xsd:complexContent>
      <xsd:extension base="CIM_EnabledLogicalElementType">
         <xsd:attribute name="OSName" type="xsd:string" use="required"/>
         <xsd:attribute name="Handle" type="xsd:string" use="required"/>
         <xsd:attribute name="Priority" type="xsd:int" use="required"/>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_UnixThreadType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ThreadType">
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

```
<xsd:complexType name="CIM_ProcessType">
   <xsd:complexContent>
      <xsd:extension base="CIM_EnabledLogicalElementType">
         <xsd:sequence>
            <xsd:element name="CIM_ServiceProcess" type="AMS_Ref"
                     minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="CIM_ProcessThread" type="CIM_ProcessThread_LinkType"
                     minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
         <xsd:attribute name="OSName" type="xsd:string" use="required"/>
         <xsd:attribute name="Handle" type="xsd:string" use="required"/>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_UnixProcessType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ProcessType">
         <xsd:sequence>
            <xsd:element name="Parameter" type="xsd:string" minOccurs="0"
                     maxOccurs="unbounded"/>
         </xsd:sequence>
         <xsd:attribute name="ProcessGroupID" type="xsd:long" use="required"/>
         <xsd:attribute name="RealUserID" type="xsd:long" use="required"/>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_ProcessThread_LinkType">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="CIM_Thread"
            type="CIM_ThreadType" />
         <xsd:element name="CIM_ThreadRef"
            type="AMS_Ref" />
         <xsd:element name="CIM_UnixThread"
            type="CIM_UnixThreadType" />
         <xsd:element name="CIM_UnixThreadRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CIM_ServiceType">
   <xsd:complexContent>
      <xsd:extension base="CIM_EnabledLogicalElementType">
            <xsd:element name="CIM_ServiceProcess" type="AMS_Ref"
                  minOccurs="0" maxOccurs="unbounded"/>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CIM_ServiceAccessPointType">
   <xsd:sequence>
      <xsd:element name="CIM_BindsToLANEndPoint" type="AMS_Ref"
         minOccurs="0" maxOccurs="unbounded" />
   </xsd:sequence>
   <xsd:attribute name="Name" type="xsd:string" use="required" />
   <xsd:attribute name="OwnerId" type="xsd:string" />
</xsd:complexType>
<xsd:complexType name="CIM_SoftwareElementType">
   <xsd:complexContent>
      <xsd:extension base="CIM_LogicalElementType">
      </xsd:extension>
```

```
      </xsd:complexContent>
   </xsd:complexType>
   <xsd:complexType name="CIM_SoftwareFeatureType">
      <xsd:complexContent>
         <xsd:extension base="CIM_LogicalElementType">
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>
   <xsd:complexType name="CIM_StorageExtentType">
      <xsd:complexContent>
         <xsd:extension base="CIM_LogicalDeviceType"></xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>
   <xsd:complexType name="CIM_SystemType">
      <xsd:complexContent>
         <xsd:extension base="CIM_EnabledLogicalElementType">
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>
   <xsd:complexType name="CIM_WatchdogType">
      <xsd:complexContent>
         <xsd:extension base="CIM_LogicalDeviceType"></xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

</xsd:schema>
```

9.6.8 Logical Hardware

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:amsm="http://www.omg.org/amsm"
   targetNamespace="http://www.omg.org/amsm"
   xmlns="http://www.omg.org/amsm">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         file AMS_LogicalHardware.xsd
         XML Schema Definition for namespace AMS Management
         Copyright (c) 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
               Themis Computer and ProgenySystems Corporation
      </xsd:documentation>
   </xsd:annotation>
   <xsd:include schemaLocation="AMS_Util.xsd" />
   <xsd:include schemaLocation="AMS_CIM.xsd" />
   <xsd:include schemaLocation="AMS_LogicalHardwareSpecification.xsd" />
   <xsd:include schemaLocation="AMS_SupportedApplicationModel.xsd" />
   <!-- types definition !-->
   <xsd:simpleType name="AMS_StdHWUtilisation">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="HU_NON_STD" />
         <xsd:enumeration value="HU_CPU_LOAD" />
         <xsd:enumeration value="HU_NETWORK_LOAD"/>
         <xsd:enumeration value="HU_NETWORK_BANDWIDTH"/>
         <xsd:enumeration value="HU_VIRTUAL_MEMORY"/>
         <xsd:enumeration value="HU_VIRTUAL_MEMORY_OCCUPATION"/>
         <xsd:enumeration value="HU_TOTAL_MEMORY"/>
         <xsd:enumeration value="HU_TOTAL_MEMORY_OCCUPATION"/>
         <xsd:enumeration value="HU_PROCESS_NUMBER"/>
         <xsd:enumeration value="HU_THREAD_NUMBER"/>
         <xsd:enumeration value="HU_DISK"/>
         <xsd:enumeration value="HU_DISK_OCCUPATION" />
```

```
   </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="CIM_HostedAccessPoint_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="CIM_ServiceAccessPoint"
            type="CIM_ServiceAccessPointType" />
         <xsd:element name="CIM_RemoteServiceAccessPoint"
            type="CIM_RemoteServiceAccessPointType" />
         <xsd:element name="CIM_NextHopRoute"
            type="CIM_NextHopRouteType" />
         <xsd:element name="CIM_NextHopIPRoute"
            type="CIM_NextHopIPRouteType" />
         <xsd:element name="CIM_ProtocolEndPoint"
            type="CIM_ProtocolEndPointType" />
         <xsd:element name="CIM_LANEndPoint"
            type="CIM_LANEndPointType" />
         <xsd:element name="CIM_IPProtocolEndPoint"
            type="CIM_IPProtocolEndPointType" />
         <xsd:element name="AMS_LANEndPoint"
            type="AMS_LANEndPointType" />
         <xsd:element name="CIM_ServiceAccessPointRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CIM_SystemComponent_LinkType"
   mixed="false">
   <xsd:sequence minOccurs="1" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="CIM_LogicalDevice"
            type="CIM_LogicalDeviceType" />
         <xsd:element name="CIM_Watchdog"
            type="CIM_WatchdogType" />
         <xsd:element name="CIM_Processor"
            type="CIM_ProcessorType" />
         <xsd:element name="CIM_Sensor" type="CIM_SensorType" />
         <xsd:element name="CIM_Display" type="CIM_DisplayType" />
         <xsd:element name="CIM_Memory" type="CIM_MemoryType" />
         <xsd:element name="CIM_PowerSupply"
            type="CIM_PowerSupplyType" />
         <xsd:element name="CIM_StorageExtent"
            type="CIM_StorageExtentType" />
         <xsd:element name="CIM_LogicalDisk"
            type="CIM_LogicalDiskType" />
         <xsd:element name="CIM_LogicalPort"
            type="CIM_LogicalPortType" />
         <xsd:element name="CIM_NetworkPort"
            type="CIM_NetworkPortType" />
         <xsd:element name="CIM_EthernetPort"
            type="CIM_EthernetPortType" />
         <xsd:element name="CIM_LogicalDeviceRef" type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CIM_InstalledOS_LinkType">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
```

```
      <xsd:choice>
         <xsd:element name="AMS_OperatingSystem"
            type="AMS_OperatingSystemType" />
         <xsd:element name="AMS_OperatingSystemRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CIM_RunningOS_LinkType">
   <xsd:sequence minOccurs="1" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="AMS_OperatingSystem"
            type="AMS_OperatingSystemType" />
         <xsd:element name="AMS_OperatingSystemRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="AMS_ComputerSystemType">
   <xsd:complexContent>
      <xsd:extension base="CIM_ComputerSystemType">
         <xsd:sequence>
            <xsd:element name="CIM_HostedAccessPoint"
               type="CIM_HostedAccessPoint_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
            <xsd:element name="CIM_SystemComponent"
               type="CIM_SystemComponent_LinkType" minOccurs="1"
               maxOccurs="unbounded" />
            <xsd:element name="CIM_SystemComponent_AdminGroup"
               type="AMS_Ref" minOccurs="0" maxOccurs="unbounded" />
            <xsd:element name="CIM_ElementLocation"
               type="AMS_Ref" minOccurs="0" maxOccurs="1" />
            <xsd:element name="CIM_InstalledOS"
               type="CIM_InstalledOS_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
            <xsd:element name="CIM_RunningOS"
               type="CIM_RunningOS_LinkType" minOccurs="1" maxOccurs="unbounded" />
            <xsd:element name="AMS_ConfSpecCS" type="AMS_Ref"
               minOccurs="0" maxOccurs="1" />
         </xsd:sequence>
         <xsd:attribute name="Name" type="xsd:string"
            use="required" />
         <xsd:attribute name="ArchitectureInfo" type="xsd:string"
            use="required" />
         <xsd:attribute name="Status" type="xsd:int"
            use="required" />
         <xsd:attribute name="NetworkLoad" type="xsd:int"
            use="required" />
         <xsd:attribute name="OwnerId" type="xsd:string" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_DomainType">
   <xsd:complexContent>
      <xsd:extension base="CIM_AdminDomainType">
         <xsd:sequence>
            <xsd:element name="AMS_DomainManagerRole"
```

```
               type="AMS_Ref" minOccurs="1" maxOccurs="unbounded" />
            <xsd:element name="CIM_SystemComponent"
               type="AMS_Ref" minOccurs="0" maxOccurs="unbounded" />
            <xsd:element name="AMS_ConfSpecDom" type="AMS_Ref"
               minOccurs="0" maxOccurs="1" />
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_DomainManagerType">
   <xsd:complexContent>
      <xsd:extension base="AMS_HostType">
         <xsd:sequence>
            <xsd:element name="AMS_DomainManagerRole"
               type="AMS_Ref" minOccurs="0" maxOccurs="1" />
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_HardwareGroupType">
   <xsd:complexContent>
      <xsd:extension base="CIM_AdminDomainType">
         <xsd:sequence>
            <xsd:element name="CIM_SystemComponent"
               type="AMS_Ref" minOccurs="0" maxOccurs="unbounded" />
            <xsd:element name="AMS_ConfSpecHG" type="AMS_Ref"
               minOccurs="0" maxOccurs="1" />
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CIM_HostedRoute_LinkType">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="CIM_NextHopRoute"
            type="CIM_NextHopRouteType" />
         <xsd:element name="CIM_NextHopRouteRef" type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="AMS_RTHU_LinkType">
   <xsd:sequence minOccurs="1" maxOccurs="1">
      <xsd:choice>
         <xsd:element name="AMS_RTHardwareUtilisation"
            type="AMS_RTHardwareUtilisationType" />
         <xsd:element name="AMS_RTHardwareUtilisationRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="AMS_RTHU_LinkType">
   <xsd:sequence minOccurs="1" maxOccurs="1">
      <xsd:choice>
         <xsd:element name="AMS_Property_StdHWUtilisation"
            type="AMS_Property_StdHWUtilisationType" />
```

```
         <xsd:element name="AMS_Property_StdHWUtilisationRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="AMS_HostType">
   <xsd:complexContent>
      <xsd:extension base="AMS_ComputerSystemType">
         <xsd:sequence>
            <xsd:element name="CIM_HostedRoute"
               type="CIM_HostedRoute_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
            <xsd:element name="AMS_RTHU"
               type="AMS_RTHU_LinkType" minOccurs="1" maxOccurs="1" />
            <xsd:element name="AMS_HostUsed" type="AMS_Ref"
               minOccurs="0" maxOccurs="unbounded" />
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_LANEndPointType">
   <xsd:complexContent>
      <xsd:extension base="CIM_LANEndPointType">
         <xsd:attribute name="Status" type="xsd:int"
            use="required" />
         <xsd:attribute name="NetworkLoad" type="xsd:int"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_OperatingSystemType">
   <xsd:complexContent>
      <xsd:extension base="CIM_OperatingSystemType">
         <xsd:sequence>
            <xsd:element name="AMS_AMSupportedByOS"
               type="AMS_Ref" minOccurs="0" maxOccurs="unbounded" />
            <xsd:element name="AMS_ConfSpecOS" type="AMS_Ref"
               minOccurs="0" maxOccurs="1" />
         </xsd:sequence>
         <xsd:attribute name="OSType" type="AMS_OSType"
            use="required" />
         <xsd:attribute name="Name" type="xsd:string"
            use="required" />
         <xsd:attribute name="Version" type="xsd:string"
            use="required" />
         <xsd:attribute name="OwnerId" type="xsd:string" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_PrinterType">
   <xsd:complexContent>
      <xsd:extension base="AMS_ComputerSystemType">
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

```
<xsd:complexType name="AMS_RTHardwareUtilisationType"
   mixed="false">
   <xsd:attribute name="CPULoad" type="xsd:int" use="required" />
   <xsd:attribute name="MemoryLoad" type="xsd:int" use="required" />
   <xsd:attribute name="DskUsage" type="xsd:int" use="required" />
</xsd:complexType>

<xsd:complexType name="AMS_Property_StdHWUtilisationType"
   mixed="false">
   <xsd:attribute name="Name" type="xsd:string" use="required" />
   <xsd:attribute name="Value" type="xsd:string" use="required"/>
   <xsd:attribute name="InstanceID" type="xsd:string" use="required" />
   <xsd:attribute name="StdName" type="AMS_StdHWUtilisation" use="required"/>
</xsd:complexType>

<xsd:complexType name="AMS_RouterType">
   <xsd:complexContent>
      <xsd:extension base="AMS_ComputerSystemType">
         <xsd:sequence>
            <xsd:element name="CIM_HostedRoute"
               type="CIM_HostedRoute_LinkType" minOccurs="0"
               maxOccurs="unbounded" />
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_SwitchType">
   <xsd:complexContent>
      <xsd:extension base="AMS_ComputerSystemType">
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<!-- structure definition !-->
<xsd:element name="AMS_LogicalHardware"
   type="AMS_LogicalHardwarePackage" />
<xsd:complexType name="AMS_LogicalHardwarePackage">
   <xsd:all minOccurs="0">
      <xsd:element name="AMS_ComputerSystem"
         type="AMS_ComputerSystemType" minOccurs="0" />
      <xsd:element name="AMS_Host" type="AMS_HostType"
         minOccurs="0" />
      <xsd:element name="AMS_Printer" type="AMS_PrinterType"
         minOccurs="0" />
      <xsd:element name="AMS_Router" type="AMS_RouterType"
         minOccurs="0" />
      <xsd:element name="AMS_Switch" type="AMS_SwitchType"
         minOccurs="0" />
      <xsd:element name="AMS_DomainManager"
         type="AMS_DomainManagerType" minOccurs="0" />

      <xsd:element name="CIM_LogicalDevice"
         type="CIM_LogicalDeviceType" minOccurs="0" />
      <xsd:element name="CIM_Watchdog" type="CIM_WatchdogType"
         minOccurs="0" />
      <xsd:element name="CIM_Processor" type="CIM_ProcessorType"
         minOccurs="0" />
      <xsd:element name="CIM_Sensor" type="CIM_SensorType"
         minOccurs="0" />
```

```
            <xsd:element name="CIM_Display" type="CIM_DisplayType"
               minOccurs="0" />
            <xsd:element name="CIM_Memory" type="CIM_MemoryType"
               minOccurs="0" />
            <xsd:element name="CIM_PowerSupply"
               type="CIM_PowerSupplyType" minOccurs="0" />
            <xsd:element name="CIM_StorageExtent"
               type="CIM_StorageExtentType" minOccurs="0" />
            <xsd:element name="CIM_LogicalPort"
               type="CIM_LogicalPortType" minOccurs="0" />
            <xsd:element name="CIM_NetworkPort"
               type="CIM_NetworkPortType" minOccurs="0" />
            <xsd:element name="CIM_EthernetPort"
               type="CIM_EthernetPortType" minOccurs="0" />
            <xsd:element name="CIM_LogicalDisk"
               type="CIM_LogicalDiskType" minOccurs="0" />

            <xsd:element name="AMS_Domain" type="AMS_DomainType"
               minOccurs="0" />
            <xsd:element name="AMS_HardwareGroup"
               type="AMS_HardwareGroupType" minOccurs="0" />
            <xsd:element name="CIM_Network" type="CIM_NetworkType"
               minOccurs="0" />

            <xsd:element name="CIM_Location" type="CIM_LocationType"
               minOccurs="0" />

            <xsd:element name="AMS_OperatingSystem"
               type="AMS_OperatingSystemType" minOccurs="0" />

            <xsd:element name="CIM_ServiceAccessPoint"
               type="CIM_ServiceAccessPointType" minOccurs="0" />
            <xsd:element name="CIM_RemoteServiceAccessPoint"
               type="CIM_RemoteServiceAccessPointType" minOccurs="0" />
            <xsd:element name="CIM_ProtocolEndPoint"
               type="CIM_ProtocolEndPointType" minOccurs="0" />
            <xsd:element name="CIM_NextHopRoute"
               type="CIM_NextHopRouteType" minOccurs="0" />
            <xsd:element name="CIM_LANEndPoint"
               type="CIM_LANEndPointType" minOccurs="0" />
            <xsd:element name="AMS_LANEndPoint"
               type="AMS_LANEndPointType" minOccurs="0" />
            <xsd:element name="CIM_IPProtocolEndPoint"
               type="CIM_IPProtocolEndPointType" minOccurs="0" />

            <xsd:element name="CIM_ConnectivityCollection"
               type="CIM_ConnectivityCollectionType" minOccurs="0" />
         </xsd:all>
      </xsd:complexType>
   </xsd:schema>
```

9.6.9 Logical Hardware Specification

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:amsm="http://www.omg.org/amsm"
   targetNamespace="http://www.omg.org/amsm"
   xmlns="http://www.omg.org/amsm">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         file AMS_LogicalHardwareSpecification.xsd
```

```
      XML Schema Definition for namespace AMS Management
      Copyright (c) 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
            Themis Computer and ProgenySystems Corporation
   </xsd:documentation>
</xsd:annotation>
<xsd:include schemaLocation="AMS_Util.xsd" />
<xsd:include schemaLocation="AMS_CIM.xsd" />
<!-- types definition !-->
<xsd:complexType name="AMS_CodedConstraintType">
   <xsd:complexContent>
      <xsd:extension base="AMS_ValueConstraintType">
         <xsd:attribute name="constraint" type="xsd:string"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_NameValue_LinkType">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="AMS_NameValueCouple"
            type="AMS_NameValueCoupleType" />
         <xsd:element name="AMS_NameValueCoupleRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="AMS_ConfigurationSpecificationType"
   mixed="false">
   <xsd:sequence>
      <xsd:element name="AMS_ConfSpecCS" type="AMS_Ref"
         minOccurs="0" maxOccurs="1" />
      <xsd:element name="AMS_ConfSpecOS" type="AMS_Ref"
         minOccurs="0" maxOccurs="1" />
      <xsd:element name="AMS_ConfSpecDom" type="AMS_Ref"
         minOccurs="0" maxOccurs="1" />
      <xsd:element name="AMS_ConfSpecHG" type="AMS_Ref"
         minOccurs="0" maxOccurs="1" />
      <xsd:element name="AMS_ConfSpecDLS" type="AMS_Ref"
         minOccurs="0" maxOccurs="1" />
      <xsd:element name="AMS_NameValue"
         type="AMS_NameValue_LinkType" minOccurs="0" maxOccurs="unbounded" />
   </xsd:sequence>
   <xsd:attribute name="InstanceID" type="xsd:string" />
</xsd:complexType>

<xsd:simpleType name="AMS_CoupleName">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="CS_NOTNORMALIZED" />
      <xsd:enumeration value="CS_NAME" />
      <xsd:enumeration value="CS_FRU" />
      <xsd:enumeration value="CS_POSITION" />
      <xsd:enumeration value="CS_INTERFACE" />
      <xsd:enumeration value="CS_MFGDATETIME" />
      <xsd:enumeration value="CS_MANUFACTURER" />
      <xsd:enumeration value="CS_PRODUCTNAME" />
      <xsd:enumeration value="CS_PRODUCTVERSION" />
      <xsd:enumeration value="CS_SERIALNUMBER" />
      <xsd:enumeration value="CS_PRODUCTTYPE" />
```

```
      <xsd:enumeration value="CS_ASSETTAG" />
      <xsd:enumeration value="CS_CHASSISTYPE" />
      <xsd:enumeration value="CS_MACADDRESS" />
      <xsd:enumeration value="CS_POWERSATE" />
      <xsd:enumeration value="CS_STATUS" />
      <xsd:enumeration value="CS_POSTRESULT" />
   </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="AMS_Constraint_LinkType">
   <xsd:sequence minOccurs="1" maxOccurs="1">
      <xsd:choice>
         <xsd:element name="AMS_ValueConstraint"
            type="AMS_ValueConstraintType" />
         <xsd:element name="AMS_RangeConstraint"
            type="AMS_RangeConstraintType" />
         <xsd:element name="AMS_SetConstraint"
            type="AMS_SetConstraintType" />
         <xsd:element name="AMS_CodedConstraint"
            type="AMS_CodedConstraintType" />
         <xsd:element name="AMS_ValueConstraintRef"
            type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="AMS_NameValueCoupleType">
   <xsd:sequence>
      <xsd:element name="AMS_Constraint"
         type="AMS_Constraint_LinkType" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
   <xsd:attribute name="StdName" type="AMS_CoupleName"
      use="required" />
   <xsd:attribute name="InstanceID" type="xsd:string" />
   <xsd:attribute name="Name" type="xsd:string" use="required" />
   <xsd:attribute name="OwnerId" type="xsd:string" />
</xsd:complexType>

<xsd:complexType name="AMS_RangeConstraintType">
   <xsd:complexContent>
      <xsd:extension base="AMS_ValueConstraintType">
         <xsd:attribute name="from" type="xsd:string"
            use="required" />
         <xsd:attribute name="to" type="xsd:string"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_SetConstraintType">
   <xsd:complexContent>
      <xsd:extension base="AMS_ValueConstraintType">
         <xsd:attribute name="set" type="xsd:string"
            use="required" />
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="AMS_ValueConstraintType">
   <xsd:attribute name="InstanceID" type="xsd:string" />
```

```
   </xsd:complexType>

   <!-- structure definition !-->
   <xsd:element name="AMS_LogicalHardwareSpecification"
      type="AMS_LogicalHardwareSpecificationPackage" />
   <xsd:complexType name="AMS_LogicalHardwareSpecificationPackage"
      mixed="false">
      <xsd:all minOccurs="0">
         <xsd:element name="AMS_ConfigurationSpecification"
            type="AMS_ConfigurationSpecificationType" minOccurs="0" />
         <xsd:element name="AMS_NameValueCouple"
            type="AMS_NameValueCoupleType" minOccurs="0" />
      </xsd:all>
   </xsd:complexType>
</xsd:schema>
```

9.6.10 Supported Application Model

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:amsm="http://www.omg.org/amsm"
   targetNamespace="http://www.omg.org/amsm"
   xmlns="http://www.omg.org/amsm">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         file AMS_SupportedApplicationModel.xsd
         XML Schema Definition for namespace AMS Management
         Copyright (c) 2005, 2006 THALES, SELEX Sistemi Integrati (SI),
               Themis Computer and ProgenySystems Corporation
      </xsd:documentation>
   </xsd:annotation>
   <xsd:include schemaLocation="AMS_Util.xsd" />
   <xsd:include schemaLocation="AMS_CIM.xsd" />
   <!-- types definition !-->
   <xsd:simpleType name="AMS_Control">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="AMS_LOAD" />
         <xsd:enumeration value="AMS_LOAD_START" />
         <xsd:enumeration value="AMS_START" />
         <xsd:enumeration value="AMS_STOP" />
         <xsd:enumeration value="AMS_HALT" />
         <xsd:enumeration value="AMS_CONTINUE" />
         <xsd:enumeration value="AMS_SHUTDOWN" />
         <xsd:enumeration value="AMS_RECOVER" />
         <xsd:enumeration value="AMS_UNLOAD" />
         <xsd:enumeration value="AMS_LOAD_DIRTY" />
         <xsd:enumeration value="LOAD_START_DIRTY" />
         <xsd:enumeration value="AMS_STOP_HALTED" />
         <xsd:enumeration value="AMS_RECLAIM" />
         <xsd:enumeration value="AMS_ALLOCATE" />
         <xsd:enumeration value="AMS_RECOVER_DIRTY" />
      </xsd:restriction>
   </xsd:simpleType>

   <xsd:simpleType name="AMS_ModelType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="AMS_PROCESS" />
         <xsd:enumeration value="AMS_J2EE" />
         <xsd:enumeration value="AMS_CCM" />
      </xsd:restriction>
   </xsd:simpleType>
```

```
<xsd:simpleType name="AMS_StdMechanism">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="MS_NONSTD" />
      <xsd:enumeration value="MS_POSIXSIGNAL" />
   </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="AMS_OSType">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="Unknown" />
      <xsd:enumeration value="Other" />
      <xsd:enumeration value="MACOS" />
      <xsd:enumeration value="ATTUNIX" />
      <xsd:enumeration value="DGUX" />
      <xsd:enumeration value="DECNT" />
      <xsd:enumeration value="Tru64 UNIX" />
      <xsd:enumeration value="OpenVMS" />
      <xsd:enumeration value="HPUX" />
      <xsd:enumeration value="AIX" />
      <xsd:enumeration value="MVS" />
      <xsd:enumeration value="OS400" />
      <xsd:enumeration value="OS/2" />
      <xsd:enumeration value="JavaVM" />
      <xsd:enumeration value="MSDOS" />
      <xsd:enumeration value="WIN3x" />
      <xsd:enumeration value="WIN95" />
      <xsd:enumeration value="WIN98" />
      <xsd:enumeration value="WINNT" />
      <xsd:enumeration value="WINCE" />
      <xsd:enumeration value="NCR3000" />
      <xsd:enumeration value="NetWare" />
      <xsd:enumeration value="OSF" />
      <xsd:enumeration value="DC/OS" />
      <xsd:enumeration value="Reliant UNIX" />
      <xsd:enumeration value="SCO UnixWare" />
      <xsd:enumeration value="SCO OpenServer" />
      <xsd:enumeration value="Sequent" />
      <xsd:enumeration value="IRIX" />
      <xsd:enumeration value="Solaris" />
      <xsd:enumeration value="SunOS" />
      <xsd:enumeration value="U6000" />
      <xsd:enumeration value="ASERIES" />
      <xsd:enumeration value="HP NonStop OS" />
      <xsd:enumeration value="HP NonStop OSS" />
      <xsd:enumeration value="BS2000" />
      <xsd:enumeration value="LINUX" />
      <xsd:enumeration value="Lynx" />
      <xsd:enumeration value="XENIX" />
      <xsd:enumeration value="VM" />
      <xsd:enumeration value="Interactive UNIX" />
      <xsd:enumeration value="BSDUNIX" />
      <xsd:enumeration value="FreeBSD" />
      <xsd:enumeration value="NetBSD" />
      <xsd:enumeration value="GNU Hurd" />
      <xsd:enumeration value="OS9" />
      <xsd:enumeration value="MACH Kernel" />
      <xsd:enumeration value="Inferno" />
      <xsd:enumeration value="QNX" />
      <xsd:enumeration value="EPOC" />
```

```
      <xsd:enumeration value="IxWorks" />
      <xsd:enumeration value="VxWorks" />
      <xsd:enumeration value="MiNT" />
      <xsd:enumeration value="BeOS" />
      <xsd:enumeration value="HP MPE" />
      <xsd:enumeration value="NextStep" />
      <xsd:enumeration value="PalmPilot" />
      <xsd:enumeration value="Rhapsody" />
      <xsd:enumeration value="Windows 2000" />
      <xsd:enumeration value="Dedicated" />
      <xsd:enumeration value="OS/390" />
      <xsd:enumeration value="VSE" />
      <xsd:enumeration value="TPF" />
      <xsd:enumeration value="Windows (R) Me" />
      <xsd:enumeration value="Caldera Open UNIX" />
      <xsd:enumeration value="OpenBSD" />
      <xsd:enumeration value="Not Applicable" />
      <xsd:enumeration value="Windows XP" />
      <xsd:enumeration value="z/OS" />
```

Issue 11519 - Vista

```
      <xsd:enumeration value="Windows Vista" />
      <xsd:enumeration value="Microsoft Windows Server 2003" />
      <xsd:enumeration
         value="Microsoft Windows Server 2003 64-Bit" />
   </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="AMS_State">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="AMS_EXECUTABLE" />
      <xsd:enumeration value="AMS_HALTED" />
      <xsd:enumeration value="AMS_LOADED" />
      <xsd:enumeration value="AMS_RUNNING" />
      <xsd:enumeration value="AMS_STOPPED" />
      <xsd:enumeration value="AMS_UNALLOCATED" />
      <xsd:enumeration value="AMS_ERROR" />
   </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="AMS_MechanismType" mixed="false">
   <xsd:attribute name="Name" type="xsd:string" use="required"/>
   <xsd:attribute name="Value" type="xsd:string" use="required"/>
   <xsd:attribute name="InstanceID" type="xsd:string" use="required"/>
   <xsd:attribute name="StdName" type="AMS_StdMechanism" use="required"/>
</xsd:complexType>

<xsd:complexType name="AMS_SupportedMechanisms_LinkType" mixed="false">
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:choice>
         <xsd:element name="AMS_Mechanism"
            type="AMS_MechanismType" />
         <xsd:element name="AMS_MechanismRef" type="AMS_Ref" />
      </xsd:choice>
   </xsd:sequence>
   <xsd:attribute name="Control" type="AMS_Control"
      use="required" />
</xsd:complexType>

<xsd:complexType name="AMS_SupportedApplicationModelType"
```

```
      mixed="false">
      <xsd:complexContent>
         <xsd:extension base="CIM_LogicalElementType">
            <xsd:sequence minOccurs="0" maxOccurs="unbounded">
               <xsd:element name="SupportedModelType"
                  type="AMS_ModelType" />
               <xsd:element name="SupportedOSType"
                  type="AMS_OSType" />
               <xsd:element name="SupportedControls"
                  type="AMS_Control" />
               <xsd:element name="SupportedStates"
                  type="AMS_State" />
               <xsd:element name="SupportedMechanisms"
                  type="AMS_SupportedMechanims_LinkType" />
            </xsd:sequence>
            <xsd:attribute name="Name" type="xsd:string"
               use="required" />
            <xsd:attribute name="ConfigurationInfo"
               type="xsd:string" use="required" />
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <!-- structure definition !-->
   <xsd:element name="AMS_SupportedApplicationModel"
      type="AMS_SupportedApplicationModelPackage" />
   <xsd:complexType name="AMS_SupportedApplicationModelPackage"
      mixed="false">
      <xsd:all minOccurs="0">
         <xsd:element name="AMS_SupportedApplicationModel"
            type="AMS_SupportedApplicationModelType" minOccurs="0" />
      </xsd:all>
   </xsd:complexType>
</xsd:schema>
```

10 DMTF CIM Managed Object Format (MOF) Platform Specific Model

10.1 Mapping Rationale

The CIM PSM presented below is provided in MOF format, the formal definition language of CIM. In general, the CIM PSM closely follows the PIM, which was also based upon CIM. The differences between the PIM and the CIM PSM are primarily due to constraints of CIM MOF and a desire to adhere to common CIM practices. One of the most notable examples is the use of ValueMaps in CIM schemas instead of enums used in the PSM. A CIM ValueMap is used in a schema instead of a separate enum as denoted in the PSM. Another restriction of CIM MOF is the lack of complex return types. When such types are used in the PSM, they are typically mapped into String return types which would then be used via reflection to access a complex object.

The CIM PSM also contains detailed comments embedded within the schemas. Such comments are typical of CIM schema definitions, and are generally derived from the descriptive text of the PSM.

10.1.1 Specific attributes and parameters information

In this paragraph, some attributes of class and parameters of operation roughly defined in the PIM, are more specified in the context of the DMTF CIM PSM.

Table 10.1 - Specific attributes for DMTF CIM PSM

Attribute	Comment
AMS_HWFilter:filter	Implementation dependent
AMS_SWFilter:filter	Implementation dependent
AMS_SupportedApplicationModel: ConfigurationInfo	Implementation dependent
AMS_ExecuteProgram:Environment	Implementation dependent
AMS_ExecuteProgram:CommandLine	Implementation dependent
AMS_ExecuteProgram:ProgramPath	Implementation dependent

Table 10.2 - Specific parameters for DMTF CIM PSM

Parameter	Comment
CreateHardwareGroup: connectivity	Implementation dependent
CreateHardwareGroup: devices	Implementation dependent
CreateHardwareGroup: resources	Implementation dependent

10.1.2 Specific data types

This paragraph specifies the data types in the context of the DMTF CIM PSM.

Table 10.3 - Data types for DMTF CIM PSM

Data type	Definition	Comment
Collection<Type>	N/A.	
datetime	datetime	
String	string	
uint8	uint8	
uint16	uint16	
uint32	utin32	
uint64	uint64	

10.1.3 Specific failure codes

Error codes have been defined in the PIM (cf. Section 7.1.13.1).

10.1.4 Mapping

10.1.4.1 Management

```
// Copyright Â© 2005, 2006 THALES, SELEX Sistemi Integrati (SI), Themis Computer
// and Progeny Systems Corporation.

// 06-08-31 - MS - Draft (2) updated to  the AMSM Revised Submission
//                 Version 1.8 - 21 August 2006
// 07-03-02 - MS - Draft (3) updated to  the AMSM Revised Submission
//                 Version 1.10 - 18 December 2006 & includes
//                 the errata from the 07-01-02 document

#pragma locale ("en_US")

// ===================================================================
// HWFilter
// ===================================================================
      [Description (" The AMS_HWFilter class models filters on "
          "hardware items. Its attribute defines the filter as:"
          "-   a string containing a regular expression on the name of"
          "    the hardware items and "
          "-   some hardware groups in the hardware must be found."
          "-   some domains in which the hardware must be found."
          "-   some networks in which network links and network "
          "    elements must be found.")]
class AMS_HWFilter {
      string filter;
```

```
};

// ==================================================================
// SWFilter
// ==================================================================
      [Description ("The AMS_SWFilter class models filters on software"
          "items. Its attribute defines the filter as:"
          "-   a string containing a regular expression on the name of"
          "    the software items and a logical expression on "
          "    attributes of ESE, appl or system (PSM dependant)"
          "-   a list of wanted states.")]
class AMS_SWFilter {
      string filter;
};

// ==================================================================
// HWManagement
// ==================================================================
       [Description ("This class is the interface to access to the "
           "logical hardware information. Its operations allow :"
           "-   to subscribe to receive periodically the state of some"
           "    logical hardwares,"
           "-   to subscribe on the change of state of some logical"
           "    hardwares,"
           "-   an access to other interfaces for a more specific view"
           "    on elements" )]
class AMS_HWManagement {

       [Description("This operation gets all the network links"
           "matching with the filter in parameter"
            "The method returns a collection of CIM_ProtocolEndPoint"
            "Since CIM/MOF spec does not support returning complex"
            "types, It returns a string type which will have to be"
            "REFLECTed into the specific return type when the class"
            "is instantiated")]
       string GetNetworkLinks(
       [IN]
         AMS_HWFilter REF filter);

       [Description("This operation gets all the networks matching "
           "with the filter in parameter"
            "The method returns a collection of CIM_Network"
            "Since CIM/MOF spec does not support returning complex"
            "types, It returns a string type which will have to be"
            "REFLECTed into the specific return type when the class"
            "is instantiated")]
       string GetNetworks(
       [IN]
         AMS_HWFilter REF filter);

       [Description("This operation gets all the computer systems"
           "matching with the filter in parameter"
            "The method returns a collection of AMS_ComputerSystem"
            "Since CIM/MOF spec does not support returning complex"
            "types, It returns a string type which will have to be"
            "REFLECTed into the specific return type when the class"
            "is instantiated")]
       string GetComputerSystems(
       [IN]
         AMS_HWFilter REF filter);
```

```
[Description("This operation gets all the hardware groups"
    "matching with the filter in parameter"
     "The method returns a collection of AMS_HardwareGroup"
     "Since CIM/MOF spec does not support returning complex"
     "types, It returns a string type which will have to be"
     "REFLECTed into the specific return type when the class"
     "is instantiated")]
string GetHardwareGroups(
[IN]
  AMS_HWFilter REF filter);

[Description("This operation gets all the domains matching "
    "with the filter in parameter"
     "The method returns a collection of AMS_Domain"
     "Since CIM/MOF spec does not support returning complex"
     "types, It returns a string type which will have to be"
     "REFLECTed into the specific return type when the class"
     "is instantiated")]
string GetDomains(
[IN]
  AMS_HWFilter REF filter);

[Description("This operation subscribes to the modifications of the"
     "load of the network elements matching with HWfilter. The data"
     "returned are a collection of AMS_RTHWIndication. This "
     "operation shall return AMS_BADFILTER if the filter parameter"
     "is wrong.")]
uint16 SubscribeNetworkLoadChange(
[IN]
  AMS_HWFilter REF filter,
[OUT]
  uint32 subscriptionID);

[Description("This operation subscribes to receive periodically the"
     "load of the network elements matching with HWfilter. The data"
     "returned are a collection of AMS_RTHWIndication. This "
     "operation shall return AMS_BADFILTER if the filter parameter"
     "is wrong.")]
uint16 SubscribeNetworkLoad (
[IN]
  uint16 delay,
[IN]
  AMS_HWFilter REF filter,
[OUT]
  uint32 subscriptionID);

[Description("This operation create a hardware group from :"
    "-   a physical location,"
    "-   subnets (parameter connectivity),"
    "-   device types (parameter devices)."
    "It shall return:"
    "-   AMS_BADCONNECTIVITY if the 'connectivity' parameter is "
    "    wrong,"
    "-   AMS_BADDEVICES if the 'devices' parameter is wrong,")]
uint16 CreateHardwareGroup (
[IN]
  CIM_Location REF location,
[IN]
  string connectivity,
```

```
        [IN]
          string devices,
        [OUT]
          AMS_HardwareGroup REF group);

        [Description ("This operation gets all locations known by the "
            "AMS service"
             "The method returns a collection of CIM_Location"
             "Since CIM/MOF spec does not support returning complex"
             "types, It returns a string type which will have to be"
             "REFLECTed into the specific return type when the class"
             "is instantiated")]
        string GetAllLocations ();

        [Description("This operation subscribes to the modifications of the"
             "status of the hardware items matching with HWfilter. The data"
             "returned are a collection of AMS_RTHWIndication. This "
             "operation shall return AMS_BADFILTER if the filter parameter"
             "is wrong.")]
        uint16 SubscribeHWStatusChange(
        [IN]
          AMS_HWFilter REF filter,
        [OUT]
          uint32 subscriptionID);

        [Description("This operation subscribes to receive periodically the"
             "status of the hardware items matching with HWfilter. The data"
             "returned are a collection of AMS_RTHWIndication. This "
             "operation shall return AMS_BADFILTER if the filter parameter"
             "is wrong.")]
        uint16 SubscribeHWStatus (
        [IN]
          uint16 delay,
        [IN]
          AMS_HWFilter REF filter,
        [OUT]
          uint32 subscriptionID);

       [Description ("This operation delete a previous subscription"
           "demand. This operation shall return AMS_BADSUBSCRIPTIONID"
           "if the parameter is erroneous")]
       uint16 Unsubscribe(
       [IN]
          uint32 subscriptionID);

       [ValueMap { "0", "1", "2", "3", "4", "5"},
       Values { "Unknown", "AMS_OK", "AMS_BADCONNECTIVITY", "AMS_BADDEVICES",
"AMS_BADFILTER",
                "AMS_BADSUBSCRIPTIONID"}]
       uint16 HWMgmt_ReturnCode;
};


// ===================================================================
// RTHWIndication
// ===================================================================
       [Description ("The AMS_RTHWIndication class gather a piece of "
           "the information brought by callbacks on hardware status."
           "Its attributes gives an AMS_ComputerSystem and an "
           "AMS_RTHardwareUtilisation. The second is the status of the"
```

```
          "first")]
class AMS_RTHWIndication : CIM_ProcessIndication {
};

// ==================================================================
// RTSWIndication
// ==================================================================
      [Description ("The AMS_RTSWIndication class gathers a piece of "
          "the information brought by callbacks on software status."
          "Its attributes gives an AMS_ExecutableSoftwareElement or an"
          "AMS_Application or an AMS_SoftwareSystem or an "
          "AMS_LoadBalancingGroup or an AMS_RedundancyGroup, and an"
          "AMS_RTSoftwareStatus. The last is the status of one of the"
          "firsts.")]
class AMS_RTSWIndication : CIM_ProcessIndication{
};

// ==================================================================
// ConfManagement
// ==================================================================
      [Description ("This class is the interface to access to the"
          "loading and unloading of configuration files")]
class AMS_ConfManagement {

      [Description ("This operation loads a configuration file")]
      string LoadConfiguration(
      [IN]
         string file);
     [Description ("This operation unloads a configuration file")]
      string UnloadConfiguration(
      [IN]
         string file);
     [Description ("This Operation returns the description of the last"
         "errot that occured with:"
         "- A string message"
         "- An errno (the errno of POSIX systems) if available,"
         "   (-1) if not"
         "- An unint16 with the (Success of Failure) error codes"
         "  per paragraph 2.3.12.1 of the spec."
         "- The element which caused the error")]
     unint16 getLastError(
      [OUT]
         string message,
      [OUT]
         uint16 errno,
      [OUT]
         unint16 errorCode,
      [OUT]
         AMS_ExecutableSoftwareElement element);
};

// ==================================================================
// DeploymentConfManagement
// ==================================================================
      [Description ("This class is the interface to access to the"
          "deployment config information. Its operations allow:"
          "-   an access to other interfaces for a more specific "
          "    view on elements.")]
class AMS_DeploymentConfManagement {
```

```
      [Description ("This operation gets all the deployment configs"
            "The method returns a collection of AMS_DeploymentConfiguration"
            "Since CIM/MOF spec does not support returning complex"
            "types, It returns a string type which will have to be"
            "REFLECTed into the specific return type when the class"
            "is instantiated")]
      string GetDeploymentConfiguration();
};

// ==================================================================
// ESEManagement
// ==================================================================
      [Description ("This class is the interface to access to the "
          "executable software element information. Its operations"
          "allow :"
          "-   to subscribe to receive periodically the state of some"
          "     executable software elements,"
          "-   to subscribe on the change of state of some executable"
          "    software elements,"
          "-   an access to other interfaces for a more specific view"
          "    on elements.")]
class AMS_ESEManagement {

      [Description ("This operation gets all the executable software"
          "elements matching with the filter in parameter."
            "The method returns a collection of"
            "AMS_ExecutableSoftwareElement"
            "Since CIM/MOF spec does not support returning complex"
            "types, It returns a string type which will have to be"
            "REFLECTed into the specific return type when the class"
            "is instantiated")]
      string GetESE (
      [IN]
        SWFilter REF filter);

      [Description ("This operation subscribes to the modifications"
          "of the status of the executable software elements matching"
          "with 'filter'. The data returned are a collection of"
          "AMS_RTHWIndication. This operation shall return"
          "AMS_BADFILTER if the filter parameter is wrong.")]
      uint16 SubscribeESEStatusChange(
      [IN]
        SWFilter REF filter,
       [OUT]
         uint32 subscriptionID);

      [Description ("This operation subscribes to receive"
          "periodically the status of the executable software "
          "elements matching with filter. The data returned"
          "are a collection of AMS_RTHWIndication. This operation "
          "shall return AMS_BADFILTER if the filter parameter is"
          "wrong.")]
       uint16 SubscribeESEStatus(
       [IN]
         uint16 delay,
       [IN]
         AMS_SWFilter REF filter,
       [OUT]
         uint32 subscriptionID);
```

```
      [Description ("This operation shuts down all the executable "
          "software elements matching with the filter given in"
          "parameter. This operation shall return AMS_BADFILTER if "
          "the filter parameter is wrong. A full explanation of the"
          "cause of the error shall be logged. ")]
      uint16 ShutdownESE(
      [IN]
         AMS_SWFilter REF filter);

      [Description ("This operation gets all the executable software"
          "elements specification."
            "The method returns a collection of AMS_ESESpec"
            "Since CIM/MOF spec does not support returning complex"
            "types, It returns a string type which will have to be"
            "REFLECTed into the specific return type when the class"
            "is instantiated")]
      string GetESESpec ();

      [Description ("This operation delete a previous subscription"
          "demand. This operation shall return AMS_BADSUBSCRIPTIONID"
          "if the parameter is erroneous")]
      uint16 Unsubscribe(
      [IN]
         uint32 subscriptionID);

      [ValueMap { "0", "1", "2", "3"},
      Values { "Unknown", "AMS_OK", "AMS_BADFILTER",
               "AMS_BADSUBSCRIPTIONID"}]
      uint16 ESEMgmt_ReturnCode;
};

// ==================================================================
// ApplicationManagement
// ==================================================================
      [Description ("This class is the interface to access to the"
          "application information. Its operations allow :"
          "-   to subscribe to receive periodically the state of some"
          "applications,"
          "-   to subscribe on the change of state of some"
          "    applications,"
          "-   an access to other interfaces for a more specific view"
          "on elements.")]
class AMS_ApplicationManagement {

      [Description ("This operation gets all the applications matching"
          "with the filter in parameter."
            "The method returns a collection of AMS_Application"
            "Since CIM/MOF spec does not support returning complex"
            "types, It returns a string type which will have to be"
            "REFLECTed into the specific return type when the class"
            "is instantiated")]
      string GetApplication (
       [IN]
         AMS_SWFilter REF filter);

      [Description ("This operation subscribes to the modifications of"
          "the status of the applications matching with filter. The"
          "data returned are a collection of AMS_RTSWIndication. This"
          "operation shall return AMS_BADFILTER if the filter"
          "parameter is wrong.")]
```

```
      uint16 SubscribeApplicationStatusChange(
       [IN]
         AMS_SWFilter REF filter,
       [OUT]
         uint32 subscriptionID);

      [Description ("This operation subscribes to receive periodically"
          "the status of the applications matching with 'filter'. The"
          "data returned are a collection of AMS_RTSWIndication. This"
          "operation shall return AMS_BADFILTER if the filter parameter"
          "is wrong.")]
      uint16 SubscribeApplicationStatus(
       [IN]
         uint16 delay,
       [IN]
         AMS_SWFilter REF filter,
       [OUT]
         uint32 subscriptionID);

      [Description ("This operation delete a previous subscription"
          "demand. This operation shall return AMS_BADSUBSCRIPTIONID"
          "if the parameter is erroneous")]
      uint16 Unsubscribe(
      [IN]
         uint32 subscriptionID);

      [ValueMap { "0", "1", "2", "3"},
      Values { "Unknown", "AMS_OK", "AMS_BADFILTER", "AMS_BADSUBSCRIPTIONID"}]
      uint16 ApplMgmt_ReturnCode;
};

// ==================================================================
// SystemManagement
// ==================================================================
      [Description ("This class is the interface to access to the"
          "system information. Its operations allow :"
          "-   to subscribe to receive periodically the state of some"
          "    systems,"
          "-   to subscribe on the change of state of some systems,"
          "-   an access to other interfaces for a more specific view"
          "    on elements.")]
class AMS_SystemManagement {

      [Description ("This operation gets all the software systems"
          "matching with the filter in parameter."
            "The method returns a collection of AMS_SoftwareSystem"
            "Since CIM/MOF spec does not support returning complex"
            "types, It returns a string type which will have to be"
            "REFLECTed into the specific return type when the class"
            "is instantiated")]
      string GetSystem(
       [IN]
         AMS_SWFilter REF filter);

      [Description ("This operation subscribes to the modifications of"
          "the status of the applications matching with filter. The"
          "data returned are a collection of AMS_RTSWIndication. This"
          "operation shall return AMS_BADFILTER if the filter"
          "parameter is wrong.")]
      uint16 SubscribeSystemStatusChange(
```

```
        [IN]
          AMS_SWFilter REF filter,
        [OUT]
          uint32 subscriptionID);

      [Description ("This operation subscribes to receive periodically"
          "the status of the software systems matching with 'filter'."
          "The data returned are a collection of AMS_RTSWIndication."
          "This operation shall return AMS_BADFILTER if the filter"
          "parameter is wrong.")]
      uint16 SubscribeSystemStatus(
       [IN]
         uint16 delay,
       [IN]
         AMS_SWFilter REF filter,
       [OUT]
         uint32 subscriptionID);

      [Description ("This operation delete a previous subscription"
          "demand. This operation shall return AMS_BADSUBSCRIPTIONID"
          "if the parameter is erroneous")]
      uint16 Unsubscribe(
      [IN]
         uint32 subscriptionID);

      [ValueMap { "0", "1", "2", "3"},
      Values { "Unknown", "AMS_OK", "AMS_BADFILTER",
               "AMS_BADSUBSCRIPTIONID"}]
      uint16 SysMgmt_ReturnCode;
};

// ===================================================================
// RedundancyGroupManagement
// ===================================================================
      [Description ("This class is the interface to access to the"
          "redundancy group information. Its operations allow :"
          "-   to subscribe to receive periodically the state of some"
          "    redundancy groups,"
          "-   to subscribe on the change of state of some redundancy"
          "    groups,"
          "-   an access to other interfaces for a more specific view"
          "    on elements.")]
class AMS_RedundancyGroupManagement {

     [Description ("This operation gets all the redundancy groups"
          "matching with the filter in parameter."
            "The method returns a collection of AMS_RedundancyGroup"
            "Since CIM/MOF spec does not support returning complex"
            "types, It returns a string type which will have to be"
            "REFLECTed into the specific return type when the class"
            "is instantiated")]
      string GetRG (
       [IN]
         AMS_SWFilter REF filter);

      [Description ("This operation subscribes to the modifications"
          "of the status of the redundancy groups matching with filter."
          "The data returned are a collection of AMS_RTSWIndication."
          "This operation shall return AMS_BADFILTER if the filter"
          "parameter is wrong.")]
```

```
      uint16 SubscribeRGStatusChange(
       [IN]
         AMS_SWFilter REF filter,
       [OUT]
         uint32 subscriptionID);

      [Description ("This operation subscribes to receive periodically"
          "the status of the redundancy groups matching with 'filter'."
          "The data returned are a collection of AMS_RTSWIndication."
          "This operation shall return AMS_BADFILTER if the filter"
          "parameter is wrong.")]
      uint16 SubscribeRGStatus (
       [IN]
         uint16 delay,
       [IN]
         AMS_SWFilter REF filter,
       [OUT]
         uint32 subscriptionID);

      [Description ("This operation delete a previous subscription"
          "demand. This operation shall return AMS_BADSUBSCRIPTIONID"
          "if the parameter is erroneous")]
      uint16 Unsubscribe(
      [IN]
         uint32 subscriptionID);

      [ValueMap { "0", "1", "2", "3"},
      Values { "Unknown", "AMS_OK", "AMS_BADFILTER",
               "AMS_BADSUBSCRIPTIONID"}]
      uint16 RGMgmt_ReturnCode;
};

// ===================================================================
// LoadBalancingManagement
// ===================================================================
      [Description ("This class is the interface to access to the"
          "load balancing group information. Its operations allow :"
          "-   to subscribe to receive periodically the state of some"
          "    load balancing groups,"
          "-   to subscribe on the change of state of some load"
          "    balancing groups,"
          "-   an access to other interfaces for a more specific view"
          "    on elements.")]
class AMS_LoadBalancingManagement {

     [Description ("This operation gets all the load balancing groups"
          "matching with the filter in parameter."
            "The method returns a collection of AMS_LoadBalancingGroup"
            "Since CIM/MOF spec does not support returning complex"
            "types, It returns a string type which will have to be"
            "REFLECTed into the specific return type when the class"
            "is instantiated")]
      string GetLB (
       [IN]
         AMS_SWFilter REF filter);

      [Description ("This operation subscribes to the modifications"
          "of the status of the load balancing groups matching with filter."
          "The data returned are a collection of AMS_RTSWIndication."
          "This operation shall return AMS_BADFILTER if the filter"
```

```
            "parameter is wrong.")]
      uint16 SubscribeLBStatusChange(
       [IN]
         AMS_SWFilter REF filter,
       [OUT]
         uint32 subscriptionID);

      [Description ("This operation subscribes to receive periodically"
          "the status of the load balancing groups matching with 'filter'."
          "The data returned are a collection of AMS_RTSWIndication."
          "This operation shall return AMS_BADFILTER if the filter"
          "parameter is wrong.")]
      uint16 SubscribeLBStatus(
       [IN]
         uint16 delay,
       [IN]
         AMS_SWFilter REF filter,
       [OUT]
         uint32 subscriptionID);

      [Description ("This operation delete a previous subscription"
          "demand. This operation shall return AMS_BADSUBSCRIPTIONID"
          "if the parameter is erroneous")]
      uint16 Unsubscribe(
      [IN]
         uint32 subscriptionID);

      [ValueMap { "0", "1", "2", "3"},
      Values { "Unknown", "AMS_OK", "AMS_BADFILTER",
               "AMS_BADSUBSCRIPTIONID"}]
      uint16 LBMgmt_ReturnCode;
};

// ===================================================================
// SAMManagement
// ===================================================================
      [Description ("This class is the interface to access to the"
          "Supported application models. Its operations allow :"
          "-   an access to other interfaces for a more specific view"
          "    on elements.")]
class AMS_SAMManagement {

      [Description ("This operation gets all the application models."
            "The method returns a collection of "
            "AMS_SupportedApplicationManagement"
            "Since CIM/MOF spec does not support returning complex"
            "types, It returns a string type which will have to be "
            "REFLECTed into the specific return type when the class"
            "is instantiated")]
      string GetAllSAM ();
};

// ===================================================================
// Log
// ===================================================================
      [Description ("The class describes the existence of the log and"
          "its characteristics. Since, there is just one log, it is a"
          "singleton. Above all, it is an interface that permits to"
          "get the administrating and consuming interfaces of the"
          "normalized lightweight logging service.")]
```

```
class AMS_Log : CIM_Log {
};

// ==================================================================
// LogRecord
// ==================================================================
      [Description ("The AMS_LogRecord class is used to instantiate"
          "records to be aggregated to a Log. It is logically "
          "equivalent with a LogRecord of the Lightweight Logging"
          "Service.")]
class AMS_LogRecord : CIM_RecordForLog {
};

// ==================================================================
// LogAdministrator, LogConsumer and LogRecord
// from the LightWeight Logging Service (LWLS)
// top level class needed to resolve the association reference with
// AMS_Log & AMS_LogRecord
// ==================================================================

class LogAdministrator {
      uint16 setMaxSize (
      [IN]
        uint32 size);

      uint16 setLogFullAction (
      [IN, Description("parameter type is class LogFullAction from the LWLS")]
        string Action);

      uint16 setAdministrativeState (
      [IN, Description("parameter type is class AdministrativeState from the LWLS")]
        string state);

      uint16  clearLog ();

      uint16 Destroy;
};

class LogConsumer {

      uint64 getRecordIdFromTime (
      [IN]
        datetime fromtime);

     [Description("Return type is class LogRecordSequence from the LWLS")]
     string retrieveById (
     [IN, OUT]
       uint64 currentId,
     [IN, OUT]
       uint16 howMany);
};

class LogRecord {

     uint64 id;
     datetime time;
};

// ==================================================================
//                               Associations
```

```
// ===================================================================

// ===================================================================
// StateFilters
// ===================================================================
      [Association, Aggregation]
class AMS_StateFilters : CIM_Component {

     [Aggregate,
      Override("GroupComponent"),
      Min(1), Max(1)]
     AMS_SWFilter REF GroupComponent;

     [Override("PartComponent")]
     AMS_State REF PartComponent;
};

// ===================================================================
// NetworkElt
// ===================================================================
      [Association]
class AMS_NetworkElt : CIM_Component {

      [Override("PartComponent"),
       Min(1), Max(1)]
      AMS_RTHWIndication REF PartComponent;

      [Override("GroupComponent"),
       Min(1), Max(1)]
      AMS_ComputerSystem REF GroupComponent;
};

// ===================================================================
// RTHS
// ===================================================================
      [Association, aggregation,
       Description("An association between"
          "AMS_Property<AMS_StdHWUtilisation, HU_NONSTD> class in order"
          "to design platform-specific hardware utilisation. Known
          "items of the AMS_StdMechanisms enumeration are:"
          "- HU_NON_STD: special value that denotes a non-normalized"
          "   value (i.e. use the "Name" attribute instead)"
          "- HU_CPU_LOAD: the corresponding value is the percentage of"
          "   CPU currently used (integer)"
          "- HU_NETWORK_LOAD: ratio between occupied Bandwidth and"
          "   available Bandwidth (float)"
          "- HU_NETWORK_BANDWIDTH: available bandwidth in bits per"
          "   second (integer)"
          "- HU_VIRTUAL_MEMORY: size in bytes of the virtual memory"
          "   (long integer)"
          "- HU_VIRTUAL_MEMORY_OCCUPATION: size in bytes of the occupied"
          "   virtual memory (long integer)"
          "- HU_TOTAL_MEMORY: size in byte of the total memory (Virtual"
          "   + Physical) (long integer)"
          "- HU_TOTAL_MEMORY_OCCUPATION: size in byte of the total"
          "   occupied memory (Virtual + Physical) (long integer)"
          "- HU_PROCESS_NUMBER: number of running processes (integer)"
          "- HU_THREAD_NUMBER: number of running threads (integer)"
          "- HU_DISK: disk size in bytes (long long integer)"
          "- HU_DISK_OCCUPATION: occupied disk size in bytes"
```

```
          "   (long long integer)")]
class AMS_RTHS : CIM_Component {

      [key, aggregate,
       Override("PartComponent"),
       Max(1)]
      AMS_RTHWIndication REF PartComponent;

      [Override("GroupComponent"),
       Min(1), Max(1)]
      AMS_Property REF GroupComponent;

      ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
                 "10", "11"},
      Values { "HU_NON_STD", "HU_CPU_LOAD", "HU_NETWORK_LOAD",
               "HU_NETWORK_BANDWIDTH", "HU_VIRTUAL_MEMORY",
               "HU_VIRTUAL_MEMORY_OCCUPATION", "HU_TOTAL_MEMORY",
               "HU_TOTAL_MEMORY_OCCUPATION", "HU_PROCESS_NUMBER",
               "HU_THREAD_NUMBER", "HU_DISK", "HU_DISK_OCCUPATION"}]
      uint16 AMSStdHWUtilisation;
};

// ===================================================================
// NetworkInFilter
// ===================================================================
      [Association]
class AMS_NetworkInFilter : CIM_Component {

      [Override("GroupComponent"),
       Min(1)]
      AMS_HWFilter REF GroupComponent;

      [Override("PartComponent"),
       Min(1)]
      CIM_Network REF PartComponent;
};

// ===================================================================
// GroupsInFilter
// ===================================================================
      [Association]
class AMS_GroupsInFilter : CIM_Component {

      [Override("GroupComponent"),
       Min(1)]
      AMS_HWFilter REF GroupComponent;

      [Override("PartComponent"),
       Min(1)]
      AMS_HardwareGroup REF PartComponent;
};

// ===================================================================
// DomainsInFilter
// ===================================================================
      [Association]
class AMS_DomainsInFilter : CIM_Component {

      [Override("GroupComponent"),
       Min(1)]
```

```
      AMS_HWFilter REF GroupComponent;

      [Override("PartComponent"),
       Min(1)]
      AMS_Domain REF PartComponent;
};

// ===================================================================
// ESE
// ===================================================================
      [Association]
class AMS_ESE : CIM_Dependency {

      [Override("Antecedent"),
       Max(1)]
      AMS_RTSWIndication REF Antecedent;

      [Override("Dependent"),
       Max(1)]
      AMS_ExecutableSoftwareElement REF Dependent;
};

// ===================================================================
// AppIndication
// ===================================================================
      [Association]
class AMS_AppIndication : CIM_Dependency {

      [Override("Antecedent"),
       Max(1)]
      AMS_RTSWIndication REF Antecedent;

      [Override("Dependent"),
       Max(1)]
      AMS_Application REF Dependent;
};

// ===================================================================
// System
// ===================================================================
      [Association]
class AMS_System : CIM_Dependency {

      [Override("Antecedent"),
       Max(1)]
      AMS_RTSWIndication REF Antecedent;

      [Override("Dependent"),
       Max(1)]
      AMS_SoftwareSystem REF Dependent;
};

// ===================================================================
// LB
// ===================================================================
      [Association]
class AMS_LB : CIM_Dependency {

      [Override("Antecedent"),
       Max(1)]
```

```
      AMS_RTSWIndication REF Antecedent;

      [Override("Dependent"),
       Max(1)]
      AMS_LoadBalancingGroup REF Dependent;
};

// ===================================================================
// RG
// ===================================================================
      [Association]
class AMS_RG : CIM_Dependency {

      [Override("Antecedent"),
       Max(1)]
      AMS_RTSWIndication REF Antecedent;

      [Override("Dependent"),
       Max(1)]
     AMS_RedundancyGroup REF Dependent;
};

// ===================================================================
// RTSW
// ===================================================================
      [Association, Aggregation]
class AMS_RTSW : CIM_Dependency {

      [Aggregate,
       Override("Antecedent"),
       Min(1), Max(1)]
      AMS_RTSWIndication REF Antecedent;

      [Override("Dependent"),
       Min(1), Max(1)]
     AMS_RTSoftwareStatus REF Dependent;
};

// ===================================================================
// AMS_LogRecordIdentity
// ===================================================================
      [Association, Aggregation]
class AMS_LogRecordIdentity : CIM_LogicalIdentity {

      [Aggregate,
       Override ( "SystemElement" ),
       Min (1), Max (1),
       Description ("The AMS Log Record")]
   AMS_LogRecord REF SystemElement;

      [Override ( "SameElement" ),
       Max (1),
       Description ("SameElement represents the additional aspects of"
          "the Lightweight Logging Service Log Record.")]
   LogRecord REF SameElement;
};


// ===================================================================
// LogAdministrator
```

```
// ==================================================================
      [Association, Aggregation]
class AMS_LogAdministrator : CIM_LogicalIdentity {

      [Key, Aggregate,
       Override("SystemElement"),
       Min(1), Max(1)]
      AMS_Log REF SystemElement;

      [Override("SameElement"),
       Min(1), Max(1),
       Description ("SameElement represents the additional aspects of"
          "the Lightweight Logging Service Log Administrator.")]
      LogAdministrator REF SameElement;
};

// ==================================================================
// LogConsumer
// ==================================================================
      [Association, Aggregation]
class AMS_LogConsumer : CIM_LogicalIdentity {

      [Key, Aggregate,
       Override("SystemElement"),
       Min(1), Max(1)]
      AMS_Log REF SystemElement;

      [Override("SameElement"),
       Min(1), Max(1),
       Description ("SameElement represents the additional aspects of"
          "the Lightweight Logging Service Log Consumer.")]
      LogConsumer REF SameElement;
};
```

10.1.4.2 Application

```
// Copyright Â© 2005, 2006 THALES, SELEX Sistemi Integrati (SI), Themis Computer
// and Progeny Systems Corporation.

// 06-08-31 - MS - Draft (2) updated to  the AMSM Revised Submission
//                 Version 1.8 - 21 August 2006
// 07-03-02 - MS - Draft (3) updated to  the AMSM Revised Submission
//                 Version 1.10 - 18 December 2006 & includes
//                 the errata from the 07-01-02 document

#pragma locale ("en_US")

// ==================================================================
// Property
// ==================================================================
      [Description ("An AMS_Property has a value and a name, which is"
          "defined by either a string or an item of an enumeration."
          "This enumeration (Value) is intended to normalize some of the"
          "possible values of the name yet letting the possibility to"
          "the implementations to define new possible names")]
class AMS_Property {

     [Key]
     string InstanceID;
```

```
     string Name;
     string Value;
};


// ==================================================================
// SoftwareSystem
// ==================================================================
      [Description ("The AMS_SoftwareSystem class models the software"
          "systems. It may be a part of an other AMS_SoftwareSystem,"
          "gathers AMS_ExecutableSoftwareElements and AMS_Applications"
          "and must be associated to the AMS_SoftwareFeatureSpec that"
          "has been used to create it and which store creation"
          "information. Its attribute are:"
          "-   a name which, with its CIM_SoftwareSystem's global"
          "      name, takes part of the make-up of its global name."
          "-   a status (started, stopped or failed)."
          "An AMS_Application is an interface that allows to start,"
          "stop and get the status of the software system itself.")]
class AMS_SoftwareSystem : CIM_ApplicationSystem {

      [Key, Override]
      string Name;

      [Description("This operation starts up the software system."
          "It must starts up all the executable software elements"
          "which belong to the system. It shall return AMS_STARTFAILED"
          "if at least one of the executable software element could not"
          "start. The state of the other executable software elements"
          "is the undefined. A full explanation of the cause of the"
          "error shall be logged")]
      uint16 Startup();

      [Description("This operation shuts down the software system."
          "It must stop all the executable software elements which"
          "belong to the system. It shall return AMS_SHUTDOWNFAILED"
          "if at least one of the executable software element could not"
          "shutdown. The state of the other executable software elements"
          "is the undefined. A full explanation of the cause of the"
          "error shall be logged")]
      uint16 Shutdown();

      [Description("The AMSRTSoftwareStatus enumerates run-time"
          "status of software items."),
      ValueMap { "0", "1", "2", "3" },
      Values { "Unknown", "SW_STARTED", "SW_STOPPED", "SW_FAILED"}]
      uint16 AMSRTSoftwareStatus;
};

// ==================================================================
// Application
// ==================================================================
      [Description ("The AMS_Application class models the"
          "applications. It is a part of an AMS_SoftwareSystem, may"
          "gather AMS_ExecutableSoftwareElements,"
          "AMS_LoadBalancingGroups and AMS_RedundancyGroups, and must"
          "be associated to the AMS_SoftwareFeatureSpec that has been"
          "used to create it and which refers to creation information"
          "Optionally, it may also gather other AMS_Applications with"
          "the association AMS_ApplicationOfApplication".
```

```
            "Its attributes are:"
            "-   a name which, with its CIM_SoftwareSystem's global"
            "      name, takes part of the make-up of its global name."
            "-   a status (started, stopped or failed)."
            "An AMS_Application is an interface that allows to start,"
            "stop and get the status of the application itself.")]
class AMS_Application : CIM_ApplicationSystem {

      [Key, Override]
      string Name;

      [Description("This operation starts up the application. It"
          "starts up all the executable software elements which belong"
          "to the application. It shall return AMS_STARTFAILED if at"
          "least one of the executable software element could not start"
          "The state of the other executable software elements is then"
          "undefined. A full explanation of the cause of the error"
          "shall be logged. After an error, the executable software"
          "elements are in an unknown state.")]
      uint16 Startup();

      [Description("This operation shuts down the application. It"
          "shuts down all the executable software elements which belong"
          "to the application. It shall return AMS_SHUTDOWNFAILED if at"
          "least one of the executable software element could not"
          "shutdown. The state of the other executable software"
          "elements is then undefined. A full explanation of the cause"
          "of the error shall be logged. After an error, the"
          "executable software elements are in an unknown state.")]
      uint16 Shutdown();


      [Description("The AMSRTSoftwareStatus enumerates run-time"
          "status of software items."),
      ValueMap { "0", "1", "2", "3" },
      Values { "Unknown", "SW_STARTED", "SW_STOPPED", "SW_FAILED"}]
      uint16 AMSRTSoftwareStatus;
};

// ===================================================================
// ExecutableSoftwareElement
// ===================================================================
      [Description ("The AMS_ExecutableSoftwareElement models a"
          "software element that is a logical element that contains"
          "the information necessary to represent and manage the"
          "functionality provided. It is part of an AMS_Application,"
          "may be associated with an AMS_RedundancyGroup and must be"
          "associated to the AMS_ESESpec that has been used to create"
          "it and which hold most of the information on the element."
          "Its attributes are:"
          "-   a name that, with its CIM_Application's or"
          "      AMS_Application's global name, takes part of the"
          "      make-up of its global name."
          "-   a current state that is taken in the states allowed by"
          "      its application model."
          "-   a status (started, stopped or failed)."
          "-   a state regarding the redundancy (no redundancy,"
          "      primary or redundant copy)"
          "An AMS_Application is an interface that allows to start,"
          "stop, load, continue and get the status of the software"
```

```
            "element itself.")]
class AMS_ExecutableSoftwareElement : CIM_Service {

      [Key, Override]
      string Name;

      [Description ("This operation starts up the executable software"
          "element with the specification actually associated by"
          "CIM_SoftwareElementServiceImplementation. It shall return:"
          "-   AMS_RESOURCEERROR if the element could not start on"
          "       account of an OS resource starvation error (lack of"
          "       memory, disk...) when performing one of the action"
          "       of the associated specification,"
          "-   AMS_RIGHTERROR if the element could not start on"
          "       account of an OS right error when performing one of"
          "       the action of the associated specification,"
          "-   AMS_BADACTION if the element could not start on account"
          "       of a badly formed action in the specification"
          "       definition,"
          "-   AMS_BADCHECK if the element could not start on account"
          "       of a badly formed check in the specification"
          "       definition,"
          "-   AMS_NOTCHECKED if at last one of the check of the"
          "       associated specification was not true,"
          "-   AMS_BADMODELTYPE if the targeted host does not support"
          "       the associated specification model type,"
          "-   AMS_STARTFAILED if the element could not start for any"
          "       other reason."
          "In all case of error, a full explanation of the cause of"
          "the error shall be logged. After an error, the executable"
          "software element is in an unknown state.")]
      uint16 Startup();

      [Description ("This operation replaces the AMS_ESESpec"
          "associated by CIM_SoftwareElementServiceImplementation,"
          "and next starts up the executable software element."
          "It shall return:"
          "-   AMS_RESOURCEERROR if the element could not start on"
          "       account of an OS resource starvation error (lack of"
          "       memory, disk...) when performing one of the action"
          "       of the associated specification,"
          "-   AMS_RIGHTERROR if the element could not start on"
          "       account of an OS right error when performing one of"
          "       the action of the associated specification,"
          "-   AMS_BADACTION if the element could not start on account"
          "       of a badly formed action in the specification"
          "       definition,"
          "-   AMS_BADCHECK if the element could not start on account"
          "       of a badly formed check in the specification"
          "       definition,"
          "-   AMS_NOTCHECKED if at last one of the check of the"
          "       associated specification was not true,"
          "-   AMS_BADMODELTYPE if the targeted host does not support"
          "       the associated specification model type,"
          "-   AMS_STARTFAILED if the element could not start for any"
          "       other reason."
          "In all case of error, a full explanation of the cause of"
          "the error shall be logged. After an error, the executable"
          "software element is in an unknown state.")]
      uint16 StartupOnSpec(
```

```
[IN]
AMS_ESESpec REF spec);

[Description ("This operation starts up the executable software"
    "element on the host and with the command line indicated by"
    "the parameters. It shall return:"
    "-   AMS_RESOURCEERROR if the element could not start on"
    "       account of an OS resource starvation error (lack of"
    "       memory, disk...) when performing one of the action"
    "       of the associated specification,"
    "-   AMS_RIGHTERROR if the element could not start on"
    "       account of an OS right error when performing one of"
    "       the action of the associated specification,"
    "-   AMS_BADACTION if the element could not start on account"
    "       of a badly formed action in the specification"
    "       definition,"
    "-   AMS_BADCHECK if the element could not start on account"
    "       of a badly formed check in the specification"
    "       definition,"
    "-   AMS_NOTCHECKED if at last one of the check of the"
    "       associated specification was not true,"
    "-   AMS_BADMODELTYPE if the targeted host does not support"
    "       the associated specification model type,"
    "-   AMS_BADCOMMANDLINE if the command line in parameter is"
    "       wrong,"
    "-   AMS_STARTFAILED if the element could not start for any"
    "       other reason."
    "In all case of error, a full explanation of the cause of"
    "the error shall be logged. After an error, the executable"
    "software element is in an unknown state.")]
uint16 StartupOnHost(
[IN]
AMS_Host REF host,
[IN]
string commandLine);

[Description("This operation shuts down the executable software"
    "element. It shall return:"
    "-   AMS_RESOURCEERROR if the element could not start on"
    "       account of an OS resource starvation error (lack of"
    "       memory, disk...) when performing one of the action"
    "       of the associated specification,"
    "-   AMS_RIGHTERROR if the element could not start on"
    "       account of an OS right error when performing one of"
    "       the action of the associated specification,"
    "-   AMS_BADACTION if the element could not start on account"
    "       of a badly formed action in the specification"
    "       definition,"
    "-   AMS_BADCHECK if the element could not start on account"
    "       of a badly formed check in the specification"
    "       definition,"
    "-   AMS_NOTCHECKED if at last one of the check of the"
    "       associated specification was not true,"
    "-   AMS_SHUTDOWNFAILED if the element could not be shut"
    "       down for any other reason."
    "In all case of error, a full explanation of the cause of"
    "the error shall be logged. After an error, the executable"
    "software element is in an unknown state.")]
uint16 Shutdown();
```

```
[Description("This operation loads the executable software"
    "element. It shall return:"
    "-   AMS_RESOURCEERROR if the element could not start on"
    "       account of an OS resource starvation error (lack of"
    "       memory, disk...) when performing one of the action"
    "       of the associated specification,"
    "-   AMS_RIGHTERROR if the element could not start on"
    "       account of an OS right error when performing one of"
    "       the action of the associated specification,"
    "-   AMS_BADMODELTYPE if the targeted host does not support"
    "       the associated specification model type,"
    "-   AMS_LOADFAILED if the element could not be start for"
    "       any other reason."
    "In all case of error, a full explanation of the cause of"
    "the error shall be logged. After an error, the executable"
    "software element is in an unknown state.")]
uint16 Load();


[Description ("This operation stops the executable software"
    "element. It shall return:"
    "-   AMS_RESOURCEERROR if the element could not be stopped"
    "       on account of an OS resource starvation error (lack"
    "       of memory, disk...) when performing one of the"
    "       action of the associated specification,"
    "-   AMS_RIGHTERROR if the element could not be stopped on"
    "       account of an OS right error when performing one of"
    "       the action of the associated specification,"
    "-   AMS_STOPFAILED if the element could not be stopped for"
    "       any other reason."
    "In all case of error, a full explanation of the cause of"
    "the error shall be logged. After an error, the executable"
    "software element is in an unknown state.")]
uint16 Stop();

[Description ("This operation continues the executable software"
    "element. It shall return:"
    "-   AMS_RESOURCEERROR if the element could not continue"
    "        on account of an OS resource starvation error (lack"
    "        of memory, disk...) when performing one of the"
    "        action of the associated specification,"
    "-   AMS_RIGHTERROR if the element could not continue on"
    "       account of an OS right error when performing one of"
    "       the action of the associated specification,"
    "-   AMS_CONTFAILED if the element could not be continued"
    "       for any other reason."
    "In all case of error, a full explanation of the cause of"
    "the error shall be logged. After an error, the executable"
    "software element is in an unknown state.")]
uint16 Continue();

[Description ("This operation forces the shutdown of the"
    "executable software element.It shall return:"
    "-   AMS_RESOURCEERROR if the element could not shutdown on"
    "       account of an OS resource starvation error (lack of"
    "       memory, disk...) when performing one of the action"
    "       of the associated specification,"
    "-   AMS_RIGHTERROR if the element could not shutdown on"
    "       account of an OS right error when performing one of"
    "       the action of the associated specification,"
```

```
            "-   AMS_SHUTDOWNFAILED if the element could not be"
            "       shutdown for any other reason."
            "In all case of error, a full explanation of the cause of"
            "the error shall be logged. After an error, the executable"
            "software element is in an unknown state.")]
        uint16 ForceShutdown();

        [Description ("This operation forces the shutdown of the"
            "executable software element.It shall return:"
            "-   AMS_ALREADYPRIMARY if the element was already a"
            "       primary element,"
            "-   AMS_NOTFT if the element is not in a redundancy"
            "       group,"
            "-   AMS_PRIMARYFAILED if the element could not be"
            "       activated for any other reason."
            "In all case of error, a full explanation of the cause of"
            "the error shall be logged. After an error, the executable"
            "software element is in an unknown state.")]
        uint16 ActivateAsPrimary();

        [Description (
            "This integer indicates the states in which executable software "
            "elements can be found  (cf Â§ 2.2.2.1)." ),
        ValueMap { "0", "1", "2", "3", "4", "5", "6" },
        Values { "Unknown", "AMS_EXECUTABLE", "AMS_HALTED", "AMS_LOADED", "AMS_RUNNING",
            "AMS_STOPPED", "AMS_UNALLOCATED" }]
        uint16 AMSState;

        [Description("The AMS_RTSoftwareStatus enumerates run-time"
            "status of software items."),
        ValueMap { "0", "1", "2", "3" },
        Values { "Unknown", "SW_STARTED", "SW_STOPPED", "SW_FAILED"}]
        uint16 AMSRTSoftwareStatus;
};

// ===================================================================
// RedundancyGroup
// ===================================================================
        [Description ("The AMS_RedundancyGroup class models the groups"
            "of elements that are working in a redundant way.."
            "It is a part of an AMS_Application, must be associated to"
            "the AMS_ExecutableElement that belongs to it, and must be"
            "associated to the AMS_SoftwareFeatureSpec that has been"
            "used to create it and which stores creation information."
            "Its attributes are:"
            "-   a name which, with its CIM_Application's global name,"
            "      takes part of the make-up of its global name. "
            "-   the style of replication (AMS_ReplicationStyle)."
            "-   a status (started, stopped or failed)."
            "An AMS_RedundancyGroup is an interface that allows to "
            "start, stop and get the status of the group itself")]
class AMS_RedundancyGroup : CIM_RedundancyGroup {

        [Key, Override]
        string Name;

        [Description ("This operation starts up the redundancy group."
            "It must starts up all the executable software elements"
            "which belong to the group. It shall return AMS_STARTFAILED"
            "if at least one of the executable software element could "
```

```
            "not start. The state of the other executable software"
            "elements is the undefined. A full explanation of the"
            "cause of the error shall be logged.")]
        uint16 Startup();

        [Description ("This operation shuts down the redundancy group."
            "It must shut down all the executable software elements"
            "which belong to the group. It shall return "
            "AMS_SHUTDOWNFAILED if at least one of the executable"
            "software element could not shutdown. The state of the"
            "other executable software elements is the undefined. A"
            "full explanation of the cause of the error shall be "
            "logged.")]
        uint16 Shutdown();

        [Description("The AMSReplicationStyle enumerates possible style"
            "for replication of a redundancy group. This enumeration"
            "comes from Fault Tolerant CORBA ([CORBA])."
            "-   RG_STATELESS: the object contains read only data, so"
            "      there is no need for recording or transferring"
            "      object's state."
            "-   RG_COLD_PASSIVE: replicas are not loaded into memory"
            "      and they only come into existence when the primary"
            "      replica fails.  Since there is only one primary"
            "      replica at any one time, the primary replica's state"
            "      must be captured in case it fails.  If the primary"
            "      replica fails, one of the cold backup replicas is"
            "      loaded into memory, and assumes the role of the new"
            "      primary replica. For the new primary replica to take"
            "      over from the old primary replica, the new replica's"
            "      state must be identical to the state of the old"
            "      primary replica. Before the new primary can fully "
            "      assume the role of the primary replica, its state is"
            "      initialized using the last checkpoint recorded"
            "      previously by the logging-recovery mechanisms."
            "-   RG_WARM_PASSIVE: this replication style differs from"
            "      the Cold Passive replication in that the state of"
            "      the primary member object of the object group gets"
            "      recorded and transferred to other member objects of"
            "      the object group (i.e. backup replicas). This type"
            "      of recovery provides faster recovery from faults"
            "      than Cold Passive. "
            "-   RG_ACTIVE and RG_ACTIVE_WITH_VOTING: with this"
            "      replication style all members of the object group"
            "      execute the invoked methods simultaneously and"
            "      expected to provide rapid recovery from faults."
            "-   RG_IMPL_DEFINED: specific implementation defined style"
            "      of replication"),
        ValueMap { "0", "1", "2", "3", "4", "5", "6"},
        Values { "Unknown", "RG_COLD_PASSIVE", "RG_WARM_PASSIVE",
                 "RG_ACTIVE", "RG_ACTIVE_WITH_VOTING", "RG_STATELESS",
                 "RG_IMPL_DEFINED"}]
        uint16 AMSReplicationStyle;

        [Description("The AMSRTSoftwareStatus class enumerates run-time"
            "status of software items."),
        ValueMap { "0", "1", "2", "3" },
        Values { "Unknown", "SW_STARTED", "SW_STOPPED", "SW_FAILED"}]
        uint16 AMSRTSoftwareStatus;
};
```

```
// ===================================================================
// LoadBalancingGroup
// ===================================================================
      [Description ("The AMS_LoadBalancingGroup class models the"
          "groups of elements that are load balanced. Load Balancing"
          "service allows to optimize the distribution of load among"
          "the available servers of the system. In this context the"
          "concept of 'Strategy' means the rule used by each"
          "application for choosing the server to execute the request"
          "within the available replicas. It is a part of an"
          "AMS_Application, must be associated to the"
          "AMS_ExecutableElements that belong to it, and must be"
          "associated to the AMS_SoftwareFeatureSpec that has been"
          "used to create it and which stores creation information."
          "Its attributes are:"
          "-   a name which, with its CIM_Application's global name,"
          "      takes part of the make-up of its global name. "
          "-   a style which is the strategy to be used for that"
          "      group."
          "-   a status (started, stopped or failed)."
          "An AMS_LoadBalancingGroup is an interface that allows to"
          "start, stop and get the status of the group itself.")]
class AMS_LoadBalancingGroup : CIM_RedundancyGroup {

      [Key, Override]
      string Name;

      [Description ("This operation starts up the load balancing group."
          "It must starts up all the executable software elements"
          "which belong to the group. It shall return AMS_STARTFAILED"
          "if at least one of the executable software element could "
          "not start. The state of the other executable software"
          "elements is the undefined. A full explanation of the"
          "cause of the error shall be logged.")]
      uint16 Startup();

      [Description ("This operation shuts down the load balancing group."
          "It must shut down all the executable software elements"
          "which belong to the group. It shall return "
          "AMS_SHUTDOWNFAILED if at least one of the executable"
          "software element could not shutdown. The state of the"
          "other executable software elements is the undefined. A"
          "full explanation of the cause of the error shall be "
          "logged.")]
      uint16 Shutdown();

      [Description("The AMS_BalancingStyle class enumerates possible"
          "style for balancing in a load balancing group. Possible"
          "values are:"
          "-   Round Robin: if a request from a client is balanced"
          "      with a Round-Robin strategy upon a group of members,"
          "      the exact order in which requests are handed over to"
          "      group members may be implementation-dependent;"
          "      however, the service must guarantee that, for a"
          "      group with n members, if a request is forwarded to a"
          "      particular group member, the next n-1 requests, from"
          "      the same client, are not forwarded to that member."
          "-   Random: requests are handed over to group members in a"
          "      randomly way."
```

```
          "-   Implementation defined: the exact order in which"
          "      requests are handed over to group members is defined"
          "      by the implementation."),
      ValueMap { "0", "1", "2", "3" },
      Values { "Unknown", "LB_ROUND_ROBIN", "LB_RANDOM",
               "LB_IMPL_DEFINED"}]
      uint16 AMSBalancingStyle;

      [Description("The AMSRTSoftwareStatus enumerates run-time"
          "status of software items."),
      ValueMap { "0", "1", "2", "3" },
      Values { "Unknown", "SW_STARTED", "SW_STOPPED", "SW_FAILED"}]
      uint16 AMSRTSoftwareStatus;
};

// ===================================================================
//                                Associations
// ===================================================================

//===================================================================
// SpecificState
// ===================================================================
      [Association, aggregation,
       Description("An association between"
          "AMS_Property<AMS_StdState, ST_NONSTD> class in order to"
          "design platform-specific states. Known items of the "
          "AMS_StdState enumeration are:"
          "- ST_NONSTD: special value that denotes a non-normalized"
          "  value (i.e. use the 'Name' attribute instead)"
          "- ST_ENV: on POSIX systems, the associated value is the"
          "  content of the environment (blank-separated strings of the"
          "  form 'var=value'")]
class AMS_SpecificState : CIM_Component {

      [Key, aggregate,
       Override("GroupComponent"),
       Max(1)]
      AMS_ExecutableSoftwareElement REF GroupComponent;

      [Override("PartComponent")]
      AMS_Property REF PartComponent;

      [ValueMap { "0", "1"},
      Values { "ST_NONSTD", "ST_ENV"}]
      uint16 AMSStdState;
};

//===================================================================
// SystemOfSystem
// ===================================================================
      [Association, Aggregation]
class AMS_SystemOfSystem : CIM_Component {

      [Key, Aggregate,
       Max(1),
       Override("GroupComponent")]
      AMS_SoftwareSystem REF GroupComponent;

      [Override("PartComponent")]
      AMS_SoftwareSystem REF PartComponent;
```

```
};

// ===================================================================
// ApplicationOfApplication
// ===================================================================
      [Association]
class AMS_ApplicationOfApplication : CIM_Component {

      [Key, Aggregate,
       Max(1),
       Override("GroupComponent")]
      AMS_Application REF GroupComponent;

      [Override("PartComponent")]
      AMS_Application REF PartComponent;
};

// ===================================================================
// ApplicationFeature
// ===================================================================
      [Association]
class AMS_ApplicationFeature : CIM_Component {

      [Override("GroupComponent")]
      AMS_Application REF GroupComponent;

      [Override("PartComponent"),
        Min(1), Max(1)]
      AMS_SoftwareFeatureSpec REF PartComponent;
};

// ===================================================================
// SystemFeature
// ===================================================================
      [Association]
class AMS_SystemFeature : CIM_Component {

      [Override("GroupComponent")]
      AMS_SoftwareSystem REF GroupComponent;

      [Override("PartComponent"),
        Max(1), Min(1)]
      AMS_SoftwareFeatureSpec REF PartComponent;
};

// ===================================================================
// RedundancyFeature
// ===================================================================
      [Association]
class AMS_RedundancyFeature : CIM_Component {

      [Override("GroupComponent")]
      AMS_RedundancyGroup REF GroupComponent;

      [Override("PartComponent"),
        Max(1), Min(1)]
      AMS_SoftwareFeatureSpec REF PartComponent;
};

// ===================================================================
```

```
// LoadBalancingFeature
// ==========================================================================
      [Association]
class AMS_LoadBalancingFeature : CIM_Component {

      [Override("GroupComponent")]
      AMS_LoadBalancingGroup REF GroupComponent;

      [Override("PartComponent"),
        Max(1), Min(1)]
      AMS_SoftwareFeatureSpec REF PartComponent;
};
```

10.1.4.3 Application Deployment

```
// Copyright Â© 2005, 2006 THALES, SELEX Sistemi Integrati (SI), Themis Computer
// and Progeny Systems Corporation.

// 06-08-31 - MS - Draft (2) updated to  the AMSM Revised Submission
//                 Version 1.8 - 21 August 2006
// 07-03-02 - MS - Draft (3) updated to  the AMSM Revised Submission
//                 Version 1.10 - 18 December 2006 & includes
//                 the errata from the 07-01-02 document

#pragma locale ("en_US")

// ==========================================================================
// DeploymentLink
// ==========================================================================
       [Description ("The AMS_DeploymentLink class models the fact"
           "that software (application, software system or software"
           "element) is running on some hosts. It belongs to an"
           "AMS_DeploymentConfiguration and is linked to some"
           "AMS_Hosts and an AMS_ExecutableSoftwareElement. If the"
           "association with an AMS_ExecutableSoftwareElement is used,"
           "just one host is allowed.")]
class AMS_DeploymentLink : CIM_LogicalElement {
       [key,
        Description ("The attribute 'LinkID' with the "
            "AMS_DeploymentConfiguration's global name, takes part of"
            "the make-up of its global name.")]
       string LinkID;
};

// ==========================================================================
// DeploymentConfiguration
// ==========================================================================
       [Description ("The AMS_DeploymentConfiguration class models the"
           "deployment configurations (run time concept). It gathers"
           "AMS_DeploymentLinks and must be associated to the"
           "AMS_DeploymentConfigurationSpec that has been used to"
           "create it and which store creation information. Its"
           "attribute is the name of the deployment configuration. An"
           " AMS_DeploymentConfiguration is an interface that allows"
           "starting or stoping a deployment configuration.")]
class AMS_DeploymentConfiguration : CIM_LogicalElement {
       [key,
        Description ("Global Name of AMS_DeploymentConfiguration")]
        string Name;
```

```
        [Description ("This operation starts up the deployment"
           "configuration: it must start up all the executable"
           "software elements which belong to the deployment"
           "configuration. It shall return AMS_STARTFAILED if at least"
           "one of the executable software element could not start."
           "The state of the other executable software elements is the"
           "undefined. A full explanation of the cause of the error"
           "shall be logged.")]
        uint16 startup ();

        [Description ("This operation shuts down the deployment "
           "configuration: it must shuts down all the executable"
           "software elements which belong to the deployment"
           "configuration. It shall return AMS_SHUTDOWNFAILED if at"
           "least one of the executable software element could not"
           "shutdown. The state of the other executable software "
           "elements is the undefined. A full explanation of the cause"
           "of the error shall be logged.")]
        uint16 shutdown();
};

// ==================================================================
//                                Associations
// ==================================================================


// ==================================================================
// DeploymentLinks
// ==================================================================
        [Association, Aggregation, Composition,
         Description ("An AMS_DeploymentConfiguration is composed of"
            "a set of AMS_DeploymentLink.  ")]
class AMS_DeploymentLinks : CIM_Component {
        [key, Aggregate, override ("GroupComponent"),
         Description ("AMS_DeploymentConfiguration is an interface that"
            "allows to start or stop a deployment configuration."),
         Min(1), Max(1)]
        AMS_DeploymentConfiguration REF GroupComponent;

        [Override ( "PartComponent" ),
         Description ("Link to the application or software system or "
            "software element on some hosts")]
        AMS_DeploymentLink REF PartComponent;
};

// ==================================================================
// DeploymentSpecAssoc
// ==================================================================
        [Association,
         Description ("Every software Configuration depends on an existing"
             "Deployment specification. NOTE: This association class was"
             "renamed from AMS_DeploymentSpec to avoid name conflict")]
class AMS_DeploymentSpecAssoc : CIM_Dependency {
        [Override("Antecedent"),
         Description ("AMS_DeploymentConfiguration is an interface that "
            "allows to start or stop a deployment configuration.")]
        AMS_DeploymentConfiguration REF Antecedent;

        [Override ("Dependent"),
         Description ("Configuration model of application or software"
```

```
            "system or software element to be deployed. See "
            "AMS_ApplicationDeploymentSpec.mof"),
         Min(1), Max(1)]
        AMS_DeploymentSpec REF Dependent;
};

// ==================================================================
// ESEDeployed
// ==================================================================
     [Association]
class AMS_ESEDeployed : CIM_Component {
     [Override("GroupComponent"),
     Min(1), Max(1)]
     AMS_DeploymentLink REF GroupComponent;

     [Override("PartComponent"),
      Min(1), Max(1)]
     AMS_ExecutableSoftwareElement REF PartComponnent;
};
```

10.1.4.4 Application Deployment Specification

```
// Copyright Â© 2005, 2006 THALES, SELEX Sistemi Integrati (SI), Themis Computer
// and Progeny Systems Corporation.

// 06-08-31 - MS - Draft (2) updated to  the AMSM Revised Submission
//                 Version 1.8 - 21 August 2006
// 07-03-02 - MS - Draft (3) updated to  the AMSM Revised Submission
//                 Version 1.10 - 18 December 2006 & includes
//                 the errata from the 07-01-02 document

#pragma locale ("en_US")

// ==================================================================
// DeploymentSpec
// ==================================================================
    [Description ("The AMS_DeploymentSpec class models the information"
        "needed to define deployment configurations. It gathers "
        "AMS_DeploymentLinkSpecs. Its attribute is its name." )]
class AMS_DeploymentSpec : CIM_LogicalElement {
    [Key, Override,
     Description ("Name of the spec.")]
    string Name;

    [Description ("This operation deploys the Executable Software Element"
        "defined in the links associated with the current deployment "
        "specification on the host defined on the same link. In case of"
        "static deployment, all associations with (ASM_Host, etc) shall be"
        "defined beforehand. In case of dynamic deployment these associations"
        "shall be determined at run time. This second case will be the"
        "subject of a future extension of the standard. It shall return:"
        "-   AMS_RESOURCEERROR if the deployment could not be performed on "
        "    account of an OS resource starvation error (lack of memory,"
        "    disk...) when performing one of the action of the associated"
        "    executable software element specifications,"
        "-   AMS_RIGHTERROR if the deployment could not be performed on "
        "    account of an OS right error when performing one of the action"
        "    of the associated executable software element specifications,"
        "-   AMS_BADACTION if the deployment could not be performed on "
        "    account of a badly formed action in the executable software"
```

```
        "    element specification definitions,"
        "-   AMS_BADCHECK if the deployment could not be performed on "
        "    account of a badly formed check in the executable software"
        "    element specification definitions,"
        "-   AMS_NOTCHECKED if at last one of the check of the associated"
        "    executable software element specifications was not true while"
        "    deploying,"
        "-   AMS_BADMODELTYPE if the targeted hosts does not support the"
        "    matching specification model types,"
        "-   AMS_DEPLOYFAILED if the element could not start for any other"
        "    reason."
        "In all case of error, a full explanation of the cause of the error"
        "shall be logged. After an error, the executable software elements"
        "already deployed are in an unknown state.")]
    uint16 Deploy();

      [ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8"},
      Values { "Unknown", "AMS_OK", "AMS_RESOURCEERROR", "AMS_RIGHTERROR",
"AMS_BADACTION",
          "AMS_BADCHECK", "AMS_NOTCHECKED", "AMS_BADMODELTYPE", "AMS_DEPLOYFAILED"}]
      uint16 Deploy_ReturnCode;
};

// ===================================================================
// DeploymentLinkSpec
// ===================================================================
    [Description ("The AMS_DeploymentLinkSpec class models the fact that"
        "a software item (application, software system or software element)"
        "will have to run on some hosts. These hosts are defined either"
        "as an actual host or with a kind of host (requested hardware)."
        "It belongs to an AMS_DeploymentSpec and is linked to:"
        "-   either some AMS_Hosts and AMS_OperatingSystem. Each host must"
        "    correspond with one operating system."
        "-   either an AMS_ESESpec or an other AMS_DeploymentLinkSpec of "
        "    which the software will be taken into account."
        "It also may have an action linked with the association class"
        "AMS_ActionOnLink, which specifies in which case (start, stop...)"
        "this action will have to be used.")]
class AMS_DeploymentLinkSpec {
     [Key,
      Description ("The ttribute 'LinkID' which, with its"
         "AMS_DeploymentSpec's global name, takes part of the make-up of"
         "its global name.")]
     string LinkID;
};


// ===================================================================
//                                Associations
// ===================================================================


// ===================================================================
// ActionOnLink
// ===================================================================
    [Association, Aggregation,
     Description ("The AMS_ActionOnLink class is an association class"
        "between AMS_DeploymentLinkSpec and CIM_Action. It is specifying"
        "in which case the action will be used: start, stop...")]
class AMS_ActionOnLink : CIM_Dependency {
```

```
    [Key, Aggregate, Override("Antecedent"),
     Max (1)]
    AMS_DeploymentLinkSpec REF Antecedent;

    [Override("Dependent"),
     Max (1)]
    CIM_Action REF Dependent;
};


// ===================================================================
// SEDeployed
// ===================================================================
     [Association]
class AMS_SEDeployed : CIM_Component {
     [Override("GroupComponent")]
     AMS_DeploymentLinkSpec REF GroupComponent;

     [Override("PartComponent"),
      Max(1)]
     AMS_ESESpec REF PartComponnent;
};

// ===================================================================
// HostUsed
// ===================================================================
     [Association]
class AMS_HostUsed : CIM_Dependency {
     [Override("Antecedent")]
     AMS_DeploymentLinkSpec REF Antecedent;

     [Overrride("Dependent")]
     AMS_Host REF Dependent;
};

// ===================================================================
// DeploymentLinkDependency
// ===================================================================
     [Association]
class AMS_DeploymentLinkDependency : CIM_Dependency {
     [Override("Antecedent")]
     AMS_DeploymentLinkSpec REF Antecedent;

     [Overrride("Dependent"),
      Max(1)]
     AMS_DeploymentLinkSpec REF Dependent;
};

// ===================================================================
// DeploymentLinks
// ===================================================================
     [Association]
class AMS_DeploymentSpecLinks : CIM_Component {
     [Key, Aggregate,
      Override("GroupComponent"),
      Max(1), Min(1)]
     AMS_DeploymentSpec REF GroupComponent;

     [Override("PartComponent")]
     AMS_DeploymentLinkSpec REF PartComponent;
```

```
};

// ==================================================================
// OSUsed
// ==================================================================
     [Association]
class AMS_OSUsed : CIM_Dependency {
     [Override("Antecedent")]
     AMS_DeploymentLinkSpec REF Antecedent;

     [Overrride("Dependent")]
     AMS_OperatingSystem REF Dependent;
};
```

10.1.4.5 Application Specification

```
// Copyright Â© 2005, 2006 THALES, SELEX Sistemi Integrati (SI), Themis Computer
// and Progeny Systems Corporation.

// 06-08-31 - MS - Draft (2) updated to  the AMSM Revised Submission
//                 Version 1.8 - 21 August 2006
// 07-03-02 - MS - Draft (3) updated to  the AMSM Revised Submission
//                 Version 1.10 - 18 December 2006 & includes
//                 the errata from the 07-01-02 document

#pragma locale ("en_US")

// ==================================================================
// SoftwareFeatureSpec
// ==================================================================
      [Description ("The AMS_SoftwareFeatureSpec class is designed as"
          "a subclass of CIM_SoftwareFeature, which is referred as "
          "-a concept [that] allows software products or application "
          "systems to be decomposed into units that have a meaning "
          "to users rather than units that reflect how the product or"
          "application was built (i.e. software elements- in CIM "
          "documentation. It may be part of another"
          "AMS_SoftwareFeatureSpec in an 'system of system' spirit."
          "It gathers AMS_ESESpecs or other AMS_SoftwareFeatureSpecs"
          "and have CIM_Checks and CIM_Actions. More specifically,"
          "actions and checks are linked to executable software "
          "element specifications with an attribute (classes"
          "AMS_SoftwareFeatureAction and AMS_SoftwareFeatureCheck),"
          "which describes in which case the actions or checks must"
          "be used : deployment, pre(post)-start, start, pre(post)-"
          "shutdown or for an alternate shutdown mechanism. The"
          "AMS_SoftwareFeatureSpec Class attribute are:"
          "-   a name that, with the global name of the"
          "    AMS_SoftwareFeatureSpec to which it belongs, takes"
          "    part of the make-up of its global name."
          "-   a type of feature (when deployed): system, application"
          "    redundancy group or load balancing group")]
class AMS_SoftwareFeatureSpec : CIM_SoftwareFeature {

      [key, override]
      string Name;

      [ValueMap { "0", "1", "2", "3", "4" },
      Values { "Unknown", "SYSTEM", "APPLICATION", "REDUNDANCY_GROUP",
               "LOADBALANCING_GROUP"}]
```

```
      uint16 AMSTypeOfFeature;
};

// ==================================================================
// ESESpec
// ==================================================================
      [Description ("The AMS_ESESpec class is designed as a subclass "
           "of CIM_SoftwareElement, which is referred in CIM "
           "documentation as -a collection of files and associated "
           "details...-. It is part of an AMS_SoftwareFeatureSpec and"
           "may be associated with AMS_Checks and AMS_Actions. More"
           "specifically, actions and checks are linked to executable"
           "software element specifications with an attribute (classes"
           "AMS_SoftwareElementAction and AMS_SoftwareElementCheck),"
           "which describes in which case the actions or checks must"
           "be used : deployment, pre(post)-start, start, "
           "pre(post)-shutdown or for an alternate shutdown mechanism"
           "The associations AMS_SEShutdownDependency and "
           "ASM_SEStartDependency amongst AMS_ESESpecs allows to "
           "define dependency graphs for, respectively, the shutdown"
           " or the start of complex applications. This association "
           "may have attributes on its link:"
           "-   TimeSinceShutdown sets a delay before the shutdown of"
           "     next the software element,"
           "-   TimeSinceStartup sets a delay before the start of"
           "     next the software element,"
           "-   CPUload sets a condition on the CPU before the start"
           "     of next the software element."
           "AMS_ESESpec Class attributes are:"
           "-   a name and a software id that, with its "
           "     AMS_SoftwareFeatureSpec's global name, takes part "
           "     of the make-up of its global name."
           "-   a model type"
           "-   a container type (ModelType)"
           "-   an optional initial redundancy state (primary,"
           "     slave... - RedInitType)")]
class AMS_ESESpec : CIM_SoftwareElement {

      [Key, Override]
      string Name;

      [Key, Override]
      string  SoftwareElementID;

      [Description ("This integer indicates the supported types of"
          "applications." ),
      ValueMap { "0", "1", "2", "3" },
      Values { "Unknown", "AMS_PROCESS", "AMS_J2EE", "AMS_CCM" }]
      uint16 AMSModelType;

      [Description ("The AMSRedundancyEltState  enumerates"
          "possible states of an executable software element regarding"
          "its role in a redundancy group:"
          "-   REDSTATE_NORG: the element is not in a redundancy group"
          "-   REDSTATE_PRIMARY: the element is the primary in its"
          "      redundancy group."
          "-   REDSTATE_PASSIVE: the element is one of the passive in"
          "     its redundancy group." ),
      ValueMap { "0", "1", "2", "3" },
      Values { "Unknown", "REDSTATE_NORG", "REDSTATE_PRIMARY", "REDSTATE_PASSIVE" }]
```

```
      uint16 AMSRedundancyEltState;
};

// ===================================================================
// ExecuteProgram
// ===================================================================
      [Description ("The AMS_ExecuteProgram is modeled as a subclass"
          "of CIM_ExecuteProgram which is defined has action that  "
          "causes programs to be executed. ExecuteProgram can be used"
          "to launch the effective software element and/or to launch "
          "a JVM and so one. (CIM documentation)."
          "The attributes are:"
          "-   CommandLine: A string that can be executed and "
          "      invokes program(s), from a system's command line"
          "      (CIM documentation)"
          "-   ProgramPath: the location or 'path' where the program"
          "      is run."
          "-   Environment: the set of environment variables to be"
          "      defined before program can be executed.")]
class AMS_ExecuteProgram : CIM_ExecuteProgram {

      string Environment;
};

// ===================================================================
// MechanizedAction
// ===================================================================
      [Description ("An AMS_MechanizedAction class works with an"
          "association to AMS_Property<AMS_StdMechanism,MS_NONSTD> in"
          "order to design the actions that use a mechanism defined"
          "by either a normalized enumeration (AMS_StdMechanism), or"
          "by an implementation-defined string, and a value.")]
class AMS_MechanizedAction : CIM_Action {
};

// ===================================================================
// CCMDeploy
// ===================================================================
      [Description ("The AMS_CCMDeploy class causes Deployment of
          "applications and systems from a component descriptor
          "(cf[D&C]) Its attribute is ComponentPackageDescriptor"
          "which is the file containing the package descriptor")]
class AMS_CCMDeploy : CIM_Action {

      string ComponentPackageDescriptor;
};

// ===================================================================
// CCMStart
// ===================================================================
      [Description ("The AMS_CCMStart class causes CCM Executable"
          "Software Element to be started")]
class AMS_CCMStart : CIM_Action {
};

// ===================================================================
// CCMStop
// ===================================================================
      [Description ("The AMS_CCMStart class causes CCM Executable"
          "Software Element to be stopped")]
```

```
class AMS_CCMStop : CIM_Action {
};

// ===================================================================
// J2EEDeploy
// ===================================================================
      [Description ("The AMS_J2EEDeploy class causes Deployment of
          "applications and systems from a J2EE archive (cf(J2EE))"
          "Its attribute is EnterpriseArchive which is"
          "the file containing the archive (.ear)")]
class AMS_J2EEDeploy : CIM_Action {

      string ComponentPackageDescriptor;
};

// ===================================================================
// J2EEStart
// ===================================================================
      [Description ("The AMS_J2EEStart class causes CCM Executable"
          "Software Element to be started")]
class AMS_J2EEStart : CIM_Action {
};

// ===================================================================
// J2EEStop
// ===================================================================
      [Description ("The AMS_J2EEStart class causes CCM Executable"
          "Software Element to be stopped")]
class AMS_J2EEStop : CIM_Action {
};

// ===================================================================
// ApplicationModelCheck
// ===================================================================
      [Description ("The AMS_SecurityCheck class models the"
          "tests in relation to an application model. One of these"
          "checks implies to verify that its application model is"
          "supported by the targeted host. Its attribute gives the"
          "AMS_SupportedApplicationModel against witch the targeted"
          "host has to be tested.")]

class AMS_ApplicationModelCheck  : CIM_Check {
};

// ===================================================================
// SecurityCheck
// ===================================================================
      [Description ("The AMS_ApplicationModelCheck class models the"
          "tests in relation to concept of security. The behavior"
          "of this class is implementation dependent ")]
class AMS_SecurityCheck  : CIM_Check {

};

// ===================================================================
//                                Associations
// ===================================================================

//===================================================================
// ActionMechanisms
```

```
// ===================================================================
      [Association, aggregation,
       Description("An association between"
```

Issue 11404 - Non-coherent naming of some items in enumerates

```
          "AMS_Property<AMS_StdMechanisms, MS_NONSTD> class in order"
          "to design platform-specific states. Known items of the "
          "AMS_StdMechanisms enumeration are:"
          "- MS_NONSTD: special value that denotes a non-normalized"
          "  value (i.e. use the 'Name' attribute instead)"
          "- MS_POSIXSIGNAL: the application model is acquainted with POSIX"
          "  signals")]
class AMS_ActionMechanisms : CIM_Component {

      [Key, aggregate,
       Override("GroupComponent"),
       Max(1)]
      AMS_MechanizedAction REF GroupComponent;

      [Override("PartComponent"),
       Max(1)]
      AMS_Property_Mechanism REF PartComponent;

      [ValueMap { "0", "1"},
      Values { "MS_NONSTD", "MS_POSIXSIGNAL"}]
      uint16 StdMechanisms;
};

// ===================================================================
// ActionCheckLink
// ===================================================================
      [Association, Abstract,
       Description ("The AMS_ActionCheckLink class is a super-class "
          "for all the association classes to CIM_Action and "
          "CIM_Check. It defines the attribute that contains the"
          "condition in which the action or the check will take"
          "place. This condition is taken among the enumeration"
          "AMSActionCheckCase.")]
class AMS_ActionCheckLink {

      [Key]
      CIM_ManagedElement REF Element;
      [key]
      CIM_ManagedElement REF CheckAction;

      [Description("The AMSActionCheckCase property is the enumeration "
          "of all the conditions in which actions or checks could "
          "take place:"
          "-   CASE_PRE_DEPLOY: just before deployment"
          "-   CASE_DEPLOY: during deployment (action: how to deploy"
          "      or check: is it allowed to deploy)"
          "-   CASE_POST_DEPLOY: just after deployment"
          "-   CASE_PRE_START: just before a startup"
          "-   CASE_START: during startup  (action: how to startup,"
          "     check: is it allowed to startup)"
          "-   CASE_POST_START: just after startup"
          "-   CASE_PRE_SHUTDOWN: just before shutdown"
          "-   CASE_SHUTDOWN: during shutdown (action: how to"
          "      shutdown, check: is it allowed to shutdown)"
          "-   CASE_POST_SHUTDOWN: just after shutdown"
```

```
          "-   CASE_ALTERNATE_SHUTDOWN: an other way to shutdown the"
          "      element. This is intended for 'last chance' killing "
          "      (such as 'kill -9' on Unix."),
      ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "10"},
      Values { "Unknown", "CASE_PRE_DEPLOY", "CASE_DEPLOY",
         "CASE_POST_DEPLOY", "CASE_PRE_START", "CASE_START",
         "CASE_POST_START", "CASE_PRE_SHUTDOWN", "CASE_SHUTDOWN",
         "CASE_POST_SHUTDOWN", "CASE_ALTERNATE_SHUTDOWN"}]
      uint16 AMSActionCheckCase;
};

// ===================================================================
// SoftwareElementCheck
// ===================================================================
      [Association,
       Description ("The AMS_SoftwareElementCheck class is an"
          "association class between AMS_ESESpec and "
          "AMS_Check. It specifies in which case the check will be "
          "used: start, stop...")]
class AMS_SoftwareElementCheck : AMS_ActionCheckLink  {

     [Override]
      AMS_ESESpec Ref Element;

      [Override]
      CIM_Check REF Check;
};

// ===================================================================
// SoftwareFeatureCheck
// ===================================================================
      [Association,
       Description ("The AMS_SoftwareFeatureCheck class is an "
          "association class between AMS_SoftwareFeature and"
          "AMS_Check. It specifies in which case the check will be"
          "used: start, stop...")]
class AMS_SoftwareFeatureCheck : AMS_ActionCheckLink  {

      [Override]
      AMS_SoftwareFeatureSpec Ref Element;

      [Override]
      CIM_Check REF Check;
};

// ===================================================================
// SoftwareElementAction
// ===================================================================
      [Association,
       Description ("The AMS_SoftwareElementAction class is an"
          "association class between AMS_ESESpec and AMS_Action. It"
          "specifies in which case the action will be used: start,"
          "stop...")]
class AMS_SoftwareElementAction : AMS_ActionCheckLink  {

     [Override]
      AMS_ESESpec Ref Element;

      [Override]
      CIM_Action REF Action;
```

```
};

// ===================================================================
// SoftwareFeatureAction
// ===================================================================
      [Association,
       Description ("The AMS_SoftwareElementAction class is an "
          "association class between AMS_SoftwareFeatureSpec and"
          "AMS_Action. It specifies in which case the action will be"
          "used: start, stop...")]
class AMS_SoftwareFeatureAction : AMS_ActionCheckLink  {

      [Override]
      AMS_SoftwareFeatureSpec Ref Element;

      [Override]
      CIM_Action REF Action;
};

// ===================================================================
// SAMCheck
// ===================================================================
      [Association]
class AMS_SAMCheck : CIM_Dependency {

      [Override("Antecedent")]
      AMS_ApplicationModelCheck REF Antecedent;

      [Override("Dependent"),
       Max(1), Min(1)]
     AMS_SupportedApplicationModel REF Dependent;
};

// ===================================================================
// SEStartDependency
// ===================================================================
      [Association,
          Description ("The AMS_SEStartDependency class is an association"
          "class among AMS_ESESpec. It defines the start dependency"
          "graph having two subclasses: on the one hand for time"
          "dependency, and on the other hand for CPU load dependency")]
class AMS_SEStartDependency : CIM_Dependency {

      [Override("Antecedent")]
      AMS_ESESpec REF Antecedent;

      [Override("Dependent"),
       Max(1)]
      AMS_ESESpec REF Dependent;

};

// ===================================================================
// SEStartTimeDepency
// ===================================================================
      [Association,
       Description ("The AMS_SEStartDependency class is an association"
          "class among AMS_ESESpec. It defines the start time"
          "dependency graph with one attribute of each links:"
          "-   TimeSinceStartup gives the delay before starting the"
```

```
          "      AMS_ESESpec.")]
class AMS_SEStartTimeDepency : AMS_SEStartDependency {

      uint16 TimeSinceStartup;

};

// ===================================================================
// SEStartCPUDependency
// ===================================================================
      [Association,
       Description ("The AMS_SEStartDependency class is an association"
          "class among AMS_ESESpec. It defines the start CPU load"
          "dependency graph with one attribute of each links:"
          "-   CPUload gives maximum value of CPU load present on"
          "the host before the executable software element is to be"
          "started.")]
class AMS_SEStartCPUDependency : AMS_SEStartDependency {

      uint16 CPUload;
};

// ===================================================================
// SEShutdownDependency
// ===================================================================
      [Association,
       Description ("The AMS_SEShutdownDependency class is an"
          "association class among AMS_ESESpec. It defines the"
          "shutdown dependency graph with one attribute of each link:"
          "-   TimeSinceShutdown gives the delay before shutting down"
          "       the AMS_ESESpec.")]
class AMS_SEShutdownDependency : CIM_Dependency {

      [Override("Antecedent"),
       Max(1)]
      AMS_ESESpec REF Antecedent;

      [Override("Dependent")]
      AMS_ESESpec REF Dependent;

      uint16 TimeSinceShutdown;
};
```

10.1.4.6 Logical Hardware

```
// Copyright Â© 2005, 2006 THALES, SELEX Sistemi Integrati (SI), Themis Computer
// and Progeny Systems Corporation.

// 06-08-31 - MS - Draft (2) updated to  the AMSM Revised Submission
//                 Version 1.8 - 21 August 2006
// 07-03-02 - MS - Draft (3) updated to  the AMSM Revised Submission
//                 Version 1.10 - 18 December 2006 & includes
//                 the errata from the 07-01-02 document

#pragma locale ("en_US")

// ===================================================================
// ComputerSystem
// ===================================================================
      [Description ("The AMS_ComputerSystem class models the hosts. "
```

```
            "Its subclasses specify which kind of elements are modeled:"
            "hosts (AMS_Host), routers (AMS_Router), switches"
            "(AMS_Switch) or printers (AMS_Printer). It aggregates"
            "some CIM_LogicalDevice which is an abstraction or"
            "emulation of a hardware entity : CIM_Processor, CIM_Memory,"
            "etc. (CIM documentation). The actual type of element is"
            "described in one of the subclasses of CIM_LogicalDevice"
            "allowed by CIM. It aggregates also some AMS_OperatingSystems"
            "as CIM_RunningOS and CIM_InstalledOS. On account of"
            "so-called virtualisation, several operating system may be"
            "running on an unique AMS_ComputerSystem, hence the"
            "multiplicity of the CIM_RunningOS association is 1..*."
            "It may be associated with a physical location "
            "(CIM_Location), with some network end points"
            "(CIM_ProtocolEndPoint). The CIM_ProtocolEndPoint class do"
            "not preclude any type of protocol. The actual protocol may"
            "be specified with one of its subclasses. For instance the"
            "CIM_IPProtocolEndPoint defines end points running on IP"
            "with IPv4 or IPv6 adresses and a subnet mask. An"
            "AMS_NetworkElement may also be associated with an"
            "AMS_Domain or an AMS_HardwareGroup. The attributes of"
            "the AMS_ComputerSystem are:"
            "-   a name that is the name of the host."
            "-   ArchitectureInfo that specify some textual information"
            "       on the architecture of the element.")]
class AMS_ComputerSystem : CIM_ComputerSystem {

      [Key, Override]
      string Name;

      string ArchitectureInfo;

      uint16 status;

      uint16 NetworkLoad;
};

// ==================================================================
// OperatingSystem
// ==================================================================
      [Description ("The class AMS_OperatingSystem models the"
            "operating system on hosts. Such an operating system belongs"
            "to an AMS_ComputerSystem either as the running operating"
            "system, or as one of the installed operating systems. It"
            "is associated with some supported application models (cf."
            "AMS_SupportedApplicationModel). Its attributes are:"
            "-   a name that, with its AMS_ComputerSystem's global name,"
            "      takes part of the make-up of its global name."
            "-   a version string that must include any patch"
            "      information.")]
class AMS_OperatingSystem : CIM_OperatingSystem {

      [Key, Override]
      string Name;

      [Override]
      String Version;

      [Description (
            "An integer indicating the type of OperatingSystem."),
```

```
        ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
            "10", "11", "12", "13", "14", "15", "16", "17", "18", "19",
            "20", "21", "22", "23", "24", "25", "26", "27", "28", "29",
            "30", "31", "32", "33", "34", "35", "36", "37", "38", "39",
            "40", "41", "42", "43", "44", "45", "46", "47", "48", "49",
            "50", "51", "52", "53", "54", "55", "56", "57", "58", "59",
            "60", "61", "62", "63", "64", "65", "66", "67", "68" },
        Values { "Unknown", "Other", "MACOS", "ATTUNIX", "DGUX",
            "DECNT", "Tru64 UNIX", "OpenVMS", "HPUX", "AIX",
            //10
            "MVS", "OS400", "OS/2", "JavaVM", "MSDOS", "WIN3x", "WIN95",
            "WIN98", "WINNT", "WINCE",
            //20
            "NCR3000", "NetWare", "OSF", "DC/OS", "Reliant UNIX",
            "SCO UnixWare", "SCO OpenServer", "Sequent", "IRIX",
            "Solaris",
            //30
            "SunOS", "U6000", "ASERIES", "TandemNSK", "TandemNT",
            "BS2000", "LINUX", "Lynx", "XENIX", "VM",
            //40
            "Interactive UNIX", "BSDUNIX", "FreeBSD", "NetBSD",
            "GNU Hurd", "OS9", "MACH Kernel", "Inferno", "QNX", "EPOC",
            //50
            "IxWorks", "VxWorks", "MiNT", "BeOS", "HP MPE", "NextStep",
            "PalmPilot", "Rhapsody", "Windows 2000", "Dedicated",
            //60
            "OS/390", "VSE", "TPF","Windows (R) Me", "Caldera Open UNIX",
            "OpenBSD", "Not Applicable", "Windows XP", "z/OS" },
        ModelCorrespondence {
            "CIM_OperatingSystem.OSType"}]
        uint16 AMSOSType;
};

// ===================================================================
// Host
// ===================================================================
      [Description ("The AMS_Host class models the hosts as points in"
          "a network. It is a subclass of AMS_ComputerSystem and a "
          "superclass of AMS_DomainManager. Since a host can be used"
          "for routing, it aggregates some routes (CIM_NextHopRoute)."
          "The CIM_NextHopRoute class does not preclude any type of"
          "protocol. The actual protocol may be specified with one of"
          "its subclasses. It has also an association that gives the"
          "real time hardware utilisation.")]
class AMS_Host : AMS_ComputerSystem {

};

// ===================================================================
// Router
// ===================================================================
      [Description ("The AMS_Router class models routers as points in"
          "a network. It is a subclass of AMS_ComputerSystem. It"
          "aggregates some routes (CIM_NextHopRoute). The"
          "CIM_NextHopRoute class do not preclude any type of protocol"
          "The actual protocol may be specified with one of its"
          "subclasses.")]
class AMS_Router : AMS_ComputerSystem {

};
```

```
// ==================================================================
// Switch
// ==================================================================
      [Description ("The AMS_Switch class models switches as points"
          "in a network. It is a subclass of AMS_ComputerSystem.")]
class AMS_Switch : AMS_ComputerSystem {

};

// ==================================================================
// HardwareGroup
// ==================================================================
      [Description ("The AMS_HardwareGroup class represents groupings"
          "of AMS_ComputerSystems. It has a set of AMS_ComputerSystem"
          "and an attribute giving its name.")]
class AMS_HardwareGroup : CIM_AdminDomain {

      [Description("This operation subscribes to the modifications of"
          "the status of the hardware items gathered in the Hardware"
          "Group. The data returned are a collection of"
          "AMS_RTHWIndication.")]
      uint16 SubscribeHWStatusChange(,
      [OUT]
      uint32 subscriptionID);

      [Description("This operation subscribes to receive periodically"
          "the status of the hardware items gathered in the Hardware"
          "Group. The data returned are a collection of"
          "AMS_RTHWIndication.")]
      uint16 SubscribeHWStatus(
      [IN]
      uint16 delay,
      [OUT]
      uint32 subscriptionID);

      [Description("This operation get the merged status computed from"
          "the status of all the Computer Systems of the Hardware"
          "Group. The actual algorithm is implementation dependant.")]
      uint16 GetMergedStatus();

      [Description("This operation subscribes to the modifications of"
          "the merged status of the Hardware Group (cf."
          "GetMergedStatus). The data returned are a collection of"
          "AMS_RTHWIndication.")]
      uint16 SubscribeMergedHWStatusChange(,
      [OUT]
      uint32 subscriptionID);

      [Description("This operation subscribes to receive periodically"
          "the merged status of the Hardware Group (cf. "
          "GetMergedStatus). The data returned are a collection of"
          "AMS_RTHWIndication.")]
      uint16 SubscribeMergedHWStatus(
      [IN]
      uint16 delay,
      [OUT]
      uint32 subscriptionID);

      [Description ("This operation delete a previous subscription"
```

```
          "demand. This operation shall return AMS_BADSUBSCRIPTIONID"
          "if the parameter is erroneous")]
      uint16 Unsubscribe(
      [IN]
         uint32 subscriptionID);

      [ValueMap { "0", "1", "2"},
      Values { "Unknown", "AMS_OK", "AMS_BADSUBSCRIPTIONID"}]
      uint16 HG_ReturnCode;
};

// ==================================================================
// Printer
// ==================================================================
      [Description ("The AMS_Printer class models printers as points"
          "in a network. It is a subclass of AMS_ComputerSystem.")]
class AMS_Printer {

};

// ==================================================================
// DomainManager
// ==================================================================
      [Description ("The AMS_DomainManager class represents a computer"
          "node in a system which is responsible for managing a"
          "AMS_Domain. There may be one or more AMS_DomainManager"
          "instances for a given AMS_Domain. AMS_DomainManager is"
          "derived from the AMS_Host class. It may be associated with"
          "an AMS_Domain.")]
class AMS_DomainManager : AMS_Host {

};

// ==================================================================
// Domain
// ==================================================================
      [Description ("The AMS_Domain class models a set of computers "
          "in a single management domain. It has a set of"
          "AMS_ComputerSystem, some AMS_DomainManagers (at least one)"
          "and an attribute giving its name.")]
class AMS_Domain : CIM_AdminDomain {

};


// ==================================================================
// LANEndPoint
// ==================================================================
      [Description ("The AMS_LANEndPoint class subclasses"
          "CIM_LANEndPoint in order to add the following attributes:"
          "-   the status of the communication end point (up or down),"
          "-   the network load on this end point (transmission rate"
          "       expressed in bytes per seconds).")]
class AMS_LANEndPoint : CIM_LANEndpoint {

      uint16 Status;
      uint16 NetworkLoad;
};

// ==================================================================
```

```
//                                Associations
// ==================================================================

// ==================================================================
// AMSupportedByOS
// ==================================================================

      [Association]
class  AMS_AMSupportedByOS : CIM_Dependency {

      [Override("Antecedent")]
      AMS_OperatingSystem REF Antecedent;

      [Override("Dependent")]
      AMS_SupportedApplicationModel REF Dependent;
};

// ==================================================================
// DomainManagerRole
// ==================================================================

      [Association]
class AMS_DomainManagerRole : CIM_Dependency {

      [Override("Antecedent"),
       Max(1)]
      AMS_Domain REF Antecedent;

      [Override("Dependent"),
       Min(1)]
      AMS_DomainManager REF Dependent;
};

// ==================================================================
// HostUsed
// ==================================================================
      [Association]
class AMS_HostUsed : CIM_Dependency {

      [Override("Antecedent")]
      AMS_DeploymentLink REF Antecedent;

      [Overrride("Dependent"),
       Max(1), Min(1)]
      AMS_Host REF Dependent;
};

// ==================================================================
// RTHU
// ==================================================================
      [Association, Aggregation,
       Description("An association between"
          "AMS_Property<AMS_StdHWUtilisation, HU_NONSTD> class in order"
          "to design platform-specific hardware utilisation. Known
          "items of the AMS_StdMechanisms enumeration are:"
          "- HU_NON_STD: special value that denotes a non-normalized"
          "   value (i.e. use the "Name" attribute instead)"
          "- HU_CPU_LOAD: the corresponding value is the percentage of"
          "   CPU currently used (integer)"
          "- HU_NETWORK_LOAD: ratio between occupied Bandwidth and"
```

```
            "   available Bandwidth (float)"
            "- HU_NETWORK_BANDWIDTH: available bandwidth in bits per"
            "   second (integer)"
            "- HU_VIRTUAL_MEMORY: size in bytes of the virtual memory"
            "   (long integer)"
            "- HU_VIRTUAL_MEMORY_OCCUPATION: size in bytes of the occupied"
            "   virtual memory (long integer)"
            "- HU_TOTAL_MEMORY: size in byte of the total memory (Virtual"
            "   + Physical) (long integer)"
            "- HU_TOTAL_MEMORY_OCCUPATION: size in byte of the total"
            "   occupied memory (Virtual + Physical) (long integer)"
            "- HU_PROCESS_NUMBER: number of running processes (integer)"
            "- HU_THREAD_NUMBER: number of running threads (integer)"
            "- HU_DISK: disk size in bytes (long long integer)"
            "- HU_DISK_OCCUPATION: occupied disk size in bytes"
            "   (long long integer)")]
class AMS_RTHU : CIM_Dependency {

      [Key, aggregate,
       Override("Antecedent"),
       Max(1), Min(1)]
      AMS_Host REF Antecedent;

      [Overrride("Dependent")]
      AMS_Property REF Dependent;

      ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
                 "10", "11"},
      Values { "HU_NON_STD", "HU_CPU_LOAD", "HU_NETWORK_LOAD",
               "HU_NETWORK_BANDWIDTH", "HU_VIRTUAL_MEMORY",
               "HU_VIRTUAL_MEMORY_OCCUPATION", "HU_TOTAL_MEMORY",
               "HU_TOTAL_MEMORY_OCCUPATION", "HU_PROCESS_NUMBER",
               "HU_THREAD_NUMBER", "HU_DISK", "HU_DISK_OCCUPATION"}]
      uint16 AMSStdHWUtilisation;
};
```

10.1.4.7 Logical Hardware Specification

```
// Copyright Â© 2005, 2006 THALES, SELEX Sistemi Integrati (SI), Themis Computer
// and Progeny Systems Corporation.

// 06-08-31 - MS - Draft (2) updated to  the AMSM Revised Submission
//                 Version 1.8 - 21 August 2006
// 07-03-02 - MS - Draft (3) updated to  the AMSM Revised Submission
//                 Version 1.10 - 18 December 2006 & includes
//                 the errata from the 07-01-02 document

#pragma locale ("en_US")


// ==================================================================
// ConfigurationSpecification
// ==================================================================

      [Description ("The AMS_ConfigurationSpecification class models"
          "the specifications of configuration for:"
          "-   operating systems (AMS_OperatingSystem),"
          "-   computers (AMS_ComputerSystem),"
          "-   network domains (AMS_Domain),"
          "-   hardware groups (AMS_HardwareGroup),"
```

```
          "-   deployment specifications (AMS_DeploymentLinkSpec)."
          "It is a set of parameters modelised by the"
          "AMS_NameValueCouple class. If no association is given for"
          "some parameter in an expected configuration, it means:"
          "Don't care.")]
class AMS_ConfigurationSpecification {

      [Key]
      string InstanceID;
};

// ===================================================================
// NameValueCouple
// ===================================================================

      [Description ("The AMS_NameValueCouple class models the"
          "parameters of a specification collection. It has a"
          "constraint on its values (association AMS_Constraint), and"
          "a name specified either as a string (attribute Name) or as"
          "an enumeration (association 'AMS_Name'). When the name is"
          "defined as a string, this name is implementation-dependant,"
          "or else it is defined in the normalized enumeration"
          "AMS_CoupleName.")]
class AMS_NameValueCouple {

     [Key]
     string InstanceID;

     string Name;

      [ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9","10",
                  "11", "12", "13", "14", "15", "16"},
      Values { "CS_NOTNORMALIZED", "CS_NAME", "CS_FRU", "CS_POSITION",
               "CS_INTERFACE", "CS_MFGDATETIME", "CS_MANUFACTURER",
               "CS_PRODUCTNAME", "CS_PRODUCTVERSION", "CS_SERIALNUMBER",
               "CS_PRODUCTTYPE", "CS_ASSETTAG", "CS_CHASSISTYPE",
               "CS_MACADDRESS", "CS_POWERSTATE", "CS_STATUS",
               "CS_POSTRESULT"}]
      uint16 AMSCoupleName;
};


// ===================================================================
// ValueConstraint
// ===================================================================

      [Description ("The AMS_ValueConstraint class models the values"
          "in a specification of configuration. It is the upper class"
          "for the real constraint class.")]
class AMS_ValueConstraint {

    [key]
    string InstanceID;
};

// ===================================================================
// CodedConstraint
// ===================================================================

      [Description ("The AMS_CodedConstraint class models contraints"
```

```
          "on the values in a specification of configuration by using"
          "an implementation-specific language.")]
class AMS_CodedConstraint : AMS_ValueConstraint {

    string Constraint;
};

// ===================================================================
// RangeConstraint
// ===================================================================

      [Description ("The AMS_RangeConstraint class models constraints"
          "on the values in a specification of configuration by"
          "specifying a lower bound (attribute from) and an upper"
          "bound (attribute to).")]
class AMS_RangeConstraint : AMS_ValueConstraint {

      [Description("Beginning of range")]
      string from;

      [Description("End of range")]
      string to;
};

// ===================================================================
// SetConstraint
// ===================================================================

      [Description ("The AMS_SetConstraint class models constraints"
          "on the values in a specification of configuration by"
          "specifying a set of possibilities.")]
class AMS_SetConstraint : AMS_ValueConstraint {

      string set;
};

// ===================================================================
//                                 Associations
// ===================================================================


//===================================================================
// NameValue
// ===================================================================
      [Association, Aggregation]
class AMS_NameValue : CIM_Component {

      [Aggregate,
       Override("GroupComponent"),
       Max(1), Min(1)]
      AMS_ConfigurationSpecification REF GroupComponent;

      [Override("PartComponent")]
      AMS_NameValueCouple REF PartComponent;
};

//===================================================================
// Constraint
// ===================================================================
      [Association, Aggregation]
```

```
class AMS_Constraint : CIM_Component {

      [Aggregate,
       Override("GroupComponent"),
       Max(1), Min(1)]
      AMS_NameValueCouple REF GroupComponent;

      [Override("PartComponent"),
       Max(1), Min(1)]
      AMS_ValueConstraint REF PartComponent;
};

//===================================================================
// ConfSpecCS
// ===================================================================
      [Association, Aggregation]
class AMS_ConfSpecCS : CIM_Component {

      [Aggregate,
       Override("GroupComponent"),
       Max(1)]
      AMS_ComputerSystem REF GroupComponent;

      [Override("PartComponent"),
       Max(1), Min(1)]
      AMS_ConfigurationSpecification REF PartComponent;
};

//===================================================================
// ConfSpecOS
// ===================================================================
      [Association, Aggregation]
class AMS_ConfSpecOS : CIM_Component {

      [Aggregate,
       Override("GroupComponent"),
       Max(1)]
      AMS_OperatingSystem REF GroupComponent;

      [Override("PartComponent"),
       Max(1), Min(1)]
      AMS_ConfigurationSpecification REF PartComponent;
};

//===================================================================
// ConfSpecDom
// ===================================================================
      [Association, Aggregation]
class AMS_ConfSpecDom : CIM_Component {

      [Aggregate,
       Override("GroupComponent"),
       Max(1)]
      AMS_Domain REF GroupComponent;

      [Override("PartComponent"),
       Max(1), Min(1)]
      AMS_ConfigurationSpecification REF PartComponent;
};
```

```
//===================================================================
// ConfSpecHG
// ===================================================================
      [Association, Aggregation]
class AMS_ConfSpecHG : CIM_Component {
      [Aggregate,
       Override("GroupComponent"),
       Max(1)]
      AMS_HardwareGroup REF GroupComponent;

      [Override("PartComponent"),
       Max(1), Min(1)]
      AMS_ConfigurationSpecification REF PartComponent;
};

//===================================================================
// ConfSpecDLS
// ===================================================================
      [Association, Aggregation]
class AMS_ConfSpecDLS : CIM_Component {

      [Aggregate,
       Override("GroupComponent"),
       Max(1)]
      AMS_DeploymentLinkSpec REF GroupComponent;

      [Override("PartComponent"),
       Max(1), Min(1)]
      AMS_ConfigurationSpecification REF PartComponent;
};
```

10.1.4.8 Supported Application Model

```
// Copyright Â© 2005, 2006 THALES, SELEX Sistemi Integrati (SI), Themis Computer
// and Progeny Systems Corporation.

// 06-08-31 - MS - Draft (2) updated to  the AMSM Revised Submission
//                 Version 1.8 - 21 August 2006
// 07-03-02 - MS - Draft (3) updated to  the AMSM Revised Submission
//                 Version 1.10 - 18 December 2006 & includes
//                 the errata from the 07-01-02 document

#pragma locale ("en_US")

// ===================================================================
// SupportedApplicationModel
// ===================================================================
      [Description (
          "The AMS_SupportedApplicationModel class specifies the"
          "application models which are supported by the AMS services i.e."
          "applications which AMS implementation can deal with. "
          "Such a model is modeled with :"
          "-    The kinds of application concerned (process or J2EE or CCM)."
          "-    The kinds of operating systems (enumeration defined in CIM)."
          "-    A collection of known control options."
          "-    A collection of known application states."
          "-    A name."
          "-    General configuration information (such as startup command"
          "      line parameters for starting ORBs)." )]
class AMS_SupportedApplicationModel : CIM_LogicalElement {
```

```
[key, override, Description("Name of the Model"),
 maxlen ( 256 )]
string Name;
[Description(" General configuration information (such as startup "
       "command line parameters for starting ORBs)." )]
string ConfigurationInfo;

[Description (
    "An integer indicating the type of OperatingSystem."),
ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
    "10", "11", "12", "13", "14", "15", "16", "17", "18", "19",
    "20", "21", "22", "23", "24", "25", "26", "27", "28", "29",
    "30", "31", "32", "33", "34", "35", "36", "37", "38", "39",
    "40", "41", "42", "43", "44", "45", "46", "47", "48", "49",
    "50", "51", "52", "53", "54", "55", "56", "57", "58", "59",
    "60", "61", "62", "63", "64", "65", "66", "67", "68" },
Values { "Unknown", "Other", "MACOS", "ATTUNIX", "DGUX",
    "DECNT", "Tru64 UNIX", "OpenVMS", "HPUX", "AIX",
    //10
    "MVS", "OS400", "OS/2", "JavaVM", "MSDOS", "WIN3x", "WIN95",
    "WIN98", "WINNT", "WINCE",
    //20
    "NCR3000", "NetWare", "OSF", "DC/OS", "Reliant UNIX",
    "SCO UnixWare", "SCO OpenServer", "Sequent", "IRIX",
    "Solaris",
    //30
    "SunOS", "U6000", "ASERIES", "TandemNSK", "TandemNT",
    "BS2000", "LINUX", "Lynx", "XENIX", "VM",
    //40
    "Interactive UNIX", "BSDUNIX", "FreeBSD", "NetBSD",
    "GNU Hurd", "OS9", "MACH Kernel", "Inferno", "QNX", "EPOC",
    //50
    "IxWorks", "VxWorks", "MiNT", "BeOS", "HP MPE", "NextStep",
    "PalmPilot", "Rhapsody", "Windows 2000", "Dedicated",
    //60
    "OS/390", "VSE", "TPF","Windows (R) Me", "Caldera Open UNIX",
    "OpenBSD", "Not Applicable", "Windows XP", "z/OS" },
ModelCorrespondence {
    "CIM_OperatingSystem.OSType",
    "AMS_SupportedApplicationModel.OtherAMSOSTypeDescription"},
ArrayType("Indexed")]
uint16 AMSOSType[];

[Description (
    "A string describing the manufacturer and OperatingSystem "
    "type - used when the OperatingSystem property, OSType, is "
    "set to 1 or 59 (\"Other\" or \"Dedicated\"). The format of "
    "the string inserted in OtherTypeDescription should be "
    "similar in format to the Values strings defined for OSType. "
    "OtherTypeDescription should be set to NULL when OSType is "
    "any value other than 1 or 59."),
 MaxLen ( 64 ),
 ModelCorrespondence { "AMS_SupportedApplicationModel.AMSOSType" },
 ArrayType("Indexed")]
 string OtherAMSOSTypeDescription[];

[Description (
   "This integer indicates the actions allowed when controlling an"
   "executable software element."),
ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
```

```
            "10", "11", "12", "13", "14", "15" },
        Values { "Unknown", "AMS_LOAD", "AMS_LOAD_START", "AMS_START", "AMS_STOP",
            "AMS_HALT", "AMS_CONTINUE", "AMS_SHUTDOWN", "AMS_RECOVER", "AMS_UNLOAD",
            "AMS_LOAD_DIRTY", "AMS_LOAD_START_DIRTY", "AMS_STOP_HALTED",
            "AMS_RECLAIM", "AMS_ALLOCATE", "AMS_RECOVER_DIRTY" },
        ArrayType("Indexed")]
        uint16 AMSControl[];

        [Description (
            "This integer indicates the states in which executable software "
            "elements can be found  (cf Â§ 2.2.2.1)." ),
        ValueMap { "0", "1", "2", "3", "4", "5", "6", "7" },
        Values { "Unknown", "AMS_EXECUTABLE", "AMS_HALTED", "AMS_LOADED", "AMS_RUNNING",
            "AMS_STOPPED", "AMS_UNALLOCATED", "AMS_ERROR"},
        ArrayType("Indexed")]
        uint16 AMSState[];

        [Description (
            "This integer indicates the supported types of applications." ),
        ValueMap { "0", "1", "2", "3" },
        Values { "Unknown", "AMS_PROCESS", "AMS_J2EE", "AMS_CCM" },
        ArrayType("Indexed")]
        uint16 AMSModelType[];
};


// ===================================================================
// Mechanism
// ===================================================================

        [Description ("The AMS_Mechanism class models possible"
            "mechanisms, such as POSIX signals, known by an application"
            "to start, stop, deploy and so forth. model"
            "It has a constraint on its value, and"
            "a name specified either as a string (attribute Name) or as"
            "an enumeration. the value of Name is only used if the"
            "StdName is MS_NONSTD, Name is implementation-dependant,"
            "or else it is defined in the normalized enumeration"
            "AMSStdMechanism.")]
class AMS_Mechanism {

        string Name;

        [ValueMap { "0", "1"},
        Values { "MS_NONSTD", "MS_POSIXSIGNAL"}]
        uint16 AMSStdMechanism;
};

// ===================================================================
//                                Associations
// ===================================================================

// ===================================================================
// SupportedMechanism
// ===================================================================
        [Association, Aggregation]
class AMS_SupportedMechanism : CIM_Component {

        [Aggregate,
         Override("GroupComponent"),
```

```
        Max(1), Min(1)]
      AMS_SupportedApplicationModel REF GroupComponent;

      [Override("PartComponent")]
      AMS_Mechanism REF PartComponent;
};
```

11 DDS/DCPS Platform Specific Model

11.1 Mapping Rationale

11.1.1 Objective

The objective of this PSM is to normalize the DDS/DCPS topics published by an AMSM service on a DDS (cf. [DDS]). The PSM is divided into two parts:

Issue 11505 - Clarification of use of DCPS PSMs w.r.t XML and CORBA PSMs

- The DCPS/f ('/f' stands for full) PSM which relates to the full DCPS PSM as described in this section. This PSM along with the XML PSM is intended to be equivalent to the CORBA/IDL and CIM profile, and thus compliant implementations are not required to deploy any elements of the CORBA and/or CIM profiles. The DCPS/f PSM shall access the static configuration information through the mandatory XML PSM.
- The DCPS/m ('/m' stands for monitoring) PSM which is a subset of the DCPS/f PSM, and contains those elements which are required for asynchronous monitoring of states of the different (software and hardware) elements. The DCPS/m PSM is defined to allow other PSMs (CORBA and/or CIM) to import it and use it for asynchronous monitoring tasks. Note that the inclusion of DCPS/m profile in CORBA and/or CIM PSM is not a mandatory, but an optional (convenience) mechanism. Whenever the DCPS/m profile is implemented along with CORBA and/or CIM PSM, the CORBA and/or CIM profile work in their usual way, independently of the DCPS/m profile.

Normative definition of both DCPS PSMs is defined in Section 11.1.5.

11.1.1.1 Topics

Issue 11517 - Root topics are not needed in PSM DCPS

There are three ‘types’ of AMSM topics:

- The “element topics” type, which represent elements of the domain and which are deemed as worthwhile within the publish/subscribe scheme of DDS:
  - supported application models
  - software systems, applications, ESEs, redundancy groups, load balancing groups
  - deployment configurations, deployment links
  - networks, protocol end points, routes, operating systems, logical devices
  - locations
  - hosts, switches, routers, printers, domain managers
  - hardware groups
  - domains
- The “indication topics” type which correspond to the subscription methods of the management classes and are roughly mapped from AMS_RTHWIndication and AMS_RTSWIndication. These topics are mainly aimed to be used when the “mixed DCPS-CORBA/IDL” way has been chosen. They are “content filtered topics” whose filters depend on the filter included in the subscription demand.

Issue 11517 - Root topics are not needed in PSM DCPS

- The "control topics" type which enables the execution of the different methods found throughout the PIM. All the “control topics” are singletons; an attribute of the topics gives the full name of the element on which the control has to be performed. The control topics include also an identifier of the request. The responses to methods are modelled as instances of topic ControlResponse, which includes the error code (return_type on the PIM level) and the request identifier (which then can be used to relate the response to the request). In case a method has output parameters other than return_type, these are obtained by reading the relevant topic. For example, an invocation of a PIM-level method CreateHardwareGroup which returns the error code and the newly created AMS_HardwareGroup, will be mapped on:
- A single instance of “control topic” CreateHardwareGroup representing the request.
- A single instance of topic ControlResponse which contains the error code of the execution of the creation and the identifier of the request.
- Creation of a new instance of topic AMS_HardwareGroup representing the newly-created hardware group.

11.1.1.2 Topics name

Each type of topic has a specific rule to define its name and type:

Issue 11518 - Add prefix to the topiic names

- The name of an “element topic” is the concatenation of "AMS_" and the name of the class of the corresponding AMSM object.
- Since an “indication topic” is aimed to be dynamically created in response to a demand of subscription, the name of an “indication topic” does not need to be normalized and, so, is implementation dependent.
- The “control topics” have a specific name given by the table below:

Issue 11517 - Root topics are not needed in PSM DCPS

Issue 11518 - Add prefix to the topiic names

Table 11.1 - Control topics

Topic Name	Comment
AMS_CT_LoadConfiguration	Loads a configuration file
AMS_CT_UnloadConfiguration	Unloads a configuration file
AMS_CT_ShutDownESE	Shuts down a filtered list of Executable Software Elements
AMS_CT_CreateHardwareGroup	Creates an hardware group
AMS_CT_ShutDown	Shuts down either a system, an application, an Executable Software Element, a load balancing group or a fault tolerance group depending of the content of the topic.
AMS_CT_StartUp	Starts either a system, an application, an Executable Software Element, a load balancing group or a fault tolerance group depending of the content of the topic.

Table 11.1 - Control topics

AMS_CT_StartUpOnHost	Starts an Executable Software Element on a host
AMS_CT_StartUpOnSpec	Starts an Executable Software Element with a given specification
AMS_CT_ActivateAsPrimary	Activates an Executable Software Element
AMS_CT_Continue	Continues an Executable Software Element
AMS_CT_ForceShutDown	Kills an Executable Software Element
AMS_CT_Load	Loads an Executable Software Element
AMS_CT_Stop	Stops an Executable Software Element
AMS_CT_ClearLog	Clear the logs
AMS_ControlResponse	The topic which contains the error codes when using preceding topics

Each control topic, apart from all input parameters, includes a unique request identifier “request_id.” All responses are modeled as instances of the ControlResponse topic, containing the identifier of the request and the return type as defined in the PIM.

11.1.1.3 Topics Key

Each type of topic has a specific rule to define its name and type:

- “Element topics” have the same keys that are defined in the PIM by the “{key}” constraints both on attributes and associations.
- The keys of an “indication topic” are located in its attributes: subscribed elements and statuses.
- “Root topics,” as singletons, have no key.
- “Control topics,” as singletons, have no key.

11.1.1.4 Topics QoS

AMSM topics share the same values for most of the DDS QoS (cf. [DDS]):

Table 11.2 - QoS of DCPS topics

QoS	Value
USER_DATA	<unspecified>
TOPIC_DATA	<unspecified>
GROUP_DATA	<unspecified>
PRESENTATION	<unspecified>
DEADLINE	Period = infinite
LATENCY_BUDGET	duration = <unspecified>
OWNERSHIP	EXCLUSIVE

Table 11.2 - QoS of DCPS topics

OWNERSHIP_STRENGTH	<unspecified>
LIVELINESS	kind = AUTOMATIC / lease_duration = <unspecified>
TIME_BASED_FILTER	<unspecified>
PARTITION	<unspecified>
TRANSPORT_PRIORITY	value=0
DESTINATION_ORDER	BY_SOURCE_TIMESTAMP
HISTORY	kind = KEEP_LAST / depth = 1
RESOURCE_LIMITS	All unlimited
ENTITY_FACTORY	<unspecified>
WRITER_DATA_LIFECYCLE	<unspecified>
READER_DATA_LIFECYCLE	<unspecified>

The other QoS (DURABILITY, RELIABILITY and LIFESPAN) will be allocated with the following principle:

- “Indication topics” have DURABILITY equals to VOLATILE, RELIABILITY equal to BEST_EFFORT and LIFESPAN.duration defined by the implementation (typically, a short lifespan is required):

DURABILITY	VOLATILE
RELIABILITY	kind = BEST_EFFORT
LIFESPAN	Implementation dependant

- As for the "Control topics" (both requests and responses), they have DURABILITY equals to VOLATILE, RELIABILITY set to RELIABLE and LIFESPAN.duration defined by the implementation:

DURABILITY	VOLATILE
RELIABILITY	kind = RELIABLE
LIFESPAN	Implementation dependant

- Others topics have DURABILITY to TRANSIENT, RELIABILITY set to RELIABLE, and LIFESPAN.duration to infinite:

DURABILITY	TRANSIENT
RELIABILITY	kind = RELIABLE
LIFESPAN	duration = infinite

11.1.1.5 Topics Data Type

11.1.1.5.1 Element Topics Data Type

Note – DDS Data types (cf. [DDS]) are defined through their equivalent CORBA/IDL structure, enumeration, and union.

The rules used to map the PIM into DCPS data types are:

- All definitions are gathered in the module “AMSM”:

```
module AMSM {
   ...
};
```

- Data types (uint16, date…) are directly mapped to DCPS types using the rules explained in 11.1.3.
- Enumeration classes are mapped to DCPS enumeration types defined as:

```
enum enumeration-name {
   an-item,
   ...
   };
```

- Non-enumeration classes are mapped to DCPS structures whose name is the name of the class:

```
struct class-name {
   ...
};
```

- Attributes of non-enumeration classes (including those of all their super classes) are next mapped with a DCPS structure element as follow:

```
attribute-type-map attribute-name ;
```

- Associations of non-enumeration classes are next mapped as follow:
- Compositions with multiplicity equal to 1 are mapped to a DCPS structure element whose name is the name of the association and type is the type of the target class of the composition (which is, in the AMSM PIM, always an enumeration class).

```
class-name assoc-name ;
```

- Compositions with multiplicity superior to 1 are mapped to a DCPS structure element whose name is the name of the association and type is a sequence of elements with the type of the target class of the composition (which is, in the AMSM PIM, always an enumeration class).

```
sequence<class-name> assoc-name ;
```

- Aggregations and simple associations are mapped to a DCPS structure element whose name is the name of the association and type is a sequence of strings. These strings contain the full names of the associated elements (cf. naming convention in 7.1.2.1). The minimum and maximum occurrences of these sequences are those expressed on the association in the PIM.
- Case minimum = maximum = 1:

```
string assoc-name ;
```

- Case maximum bounded:

```
sequence<string, maximum> assoc-name ;
```

- Case maximum unbounded:

```
sequence<string> assoc-name ;
```

- Associations whose reverse association is an aggregation or a composition are not mapped.
- Classes without any attributes or associations are not mapped.
- An attribute of type “string” and name “Owner” is added so as to design the owner topic of the topic (full name).
- Instantiations of a template class are mapped as a regular class but with a structure name built from the concatenation of the names of the template and of the classes among the template arguments.

Those rules are not intended to be general rules to map a UML PIM to DCPS data types. They are specific rules established for the AMSM specific case.

11.1.1.5.2 Element Topics Data Type Exceptions

The preceding rules deal neither with all the cases which arise in the AMSM PIM, nor with some of the requirements of DDS norm.

Hereby are those specific cases:

- Non-alphabetical and non-numerical characters in enumeration items (such as '/,' '(', ')') are mapped to underscores.
- Some associations such as CIM_SystemComponent or CIM_RedundancyComponent appear two or three times on a same class, which leads to multiple structure elements with the same name. Since this is not allowed in a DCPS data type, those multiple definitions are gathered in one.
- Elements that are not mapped:
- Management and filter classes
- Specification classes (Application Specification, Software Specification).
- Classes related to logging
- Virtual classes: AMS_ComputerSystem, CIM_LogicalDevice
- Association classes are not properly mapped, yet none of them need to be mapped in the actual AMSM PIM.
- Specific structures aimed to contain global lists of "root" elements are added: supported application models, software systems, deployments configurations, networks, locations, hosts, switches, routers, printers, domain managers, domains, and hardware groups.

11.1.1.5.3 Indication Topics Data Type

Indication topics data type are defined by as follow:

- Software indications are designed as an AMS_RTSWIndication structure which associates a software element (to be chosen amongst Executable Software Elements, Applications, Systems and groups) with its status and a time.

```
//
union AMS_RTSWIndication_Element switch (AMS_RTSWIndication_Case) {
   case AMS_ESE_IndicationCase:
      // full name of elements of AMS_ExecutableSoftwareElement
      string ESE;
   case AMS_Application_IndicationCase:
      // full name of elements of AMS_Application
      string Application;
   case AMS_System_IndicationCase:
      // full name of elements of AMS_SoftwareSystem
      string System;
   case AMS_LB_IndicationCase:
      // full name of elements of AMS_LoadBalancingGroup
      string LB;
   case AMS_FT_IndicationCase:
      // full name of elements of AMS_RedundancyGroup
      string RG;
};
struct AMS_RTSWIndication {
   //
```

```
    AMS_RTSWIndication_Case ElementCase;
    AMS_RTSWIndication_Element Element;
    //
    AMS_RTSoftwareStatus AMS_RTSW;
    //
    string IndicationTime;
};
```

Hardware indications are designed as an AMS_RTHWIndication structure which associates a network element (to be chosen among the AMS_ComputerSystem and its subclasses) with its platform-specific hardware utilizations (AMS_Property<AMS_StdHWUtilisation, HU_NONSTD>).

```
//
struct AMS_RTHWIndication {
    // the full name of an element of AMS_ComputerSystem
    string AMS_NetworkElt;
    // AMS_Property<AMS_StdHWUtilisation, HU_NONSTD>
    sequence<AMS_PropertyStdHWUtilisation> AMS_RTHS;
    //
    string IndicationTime;
};
```

Issue 11517 - Root topics are not needed in PSM DCPS

11.1.1.5.4 Control Topics Data Type

Issue 11518 - Add prefix to the topic names

“Control topics” have specific data types which names are the same as the topic names (cf. table in paragraph 11.1.1.2).

Each "Control topic" except "AMS_ControlResponse" has its own data type which gathers:

- the identifier of the instance (request)
- the full name (string) of the elements on which the control is performed
- an attribute per parameter of the corresponding method

AMS_ControlResponse has the following datatype which gathers all the information resulting from a call to the GetLastError method:

Issue 11518 - Add prefix to the topic names

```
struct AMS_ErrorStruct {
// message
string Message;
// errno
long Number
// error code
AMS_ErrorCode Code;
// full path to the element
string Element
};

struct AMS_ControlResponse {
// identifier of the request
long request_id;
// the ErrorStruct
AMS_ErrorStruct Error;
```

```
}

#pragma keylist AMS_ControlResponse request_id
```

11.1.2 Specific attributes and parameters information

In this section, some attributes of class and parameters of operation, which are roughly defined in the PIM, are more specifically defined in the context of the DCPS PSM.

Table 11.3 - Specific attributes for DCPS PSM

Attribute	Comment
AMS_HWFilter:filter	Implementation dependent
AMS_SWFilter:filter	Implementation dependent
AMS_SupportedApplicationModel: ConfigurationInfo	Implementation dependent
AMS_ExecuteProgram:Environment	Implementation dependent
AMS_ExecuteProgram:CommandLine	Implementation dependent
AMS_ExecuteProgram:ProgramPath	Implementation dependent

Table 11.4 - Specific parameters for DCPS PSM

Parameter	Comment
CreateHardwareGroup: connectivity	N/A
CreateHardwareGroup: devices	N/A
CreateHardwareGroup: resources	N/A

11.1.3 Specific data types

This paragraph specifies the data types in the context of the DCPS PSM.

Table 11.5 - Data types for DCPS PSM

Data type	Definition	Comment
Collection<Type>	N/A	Since these data types are encountered in parameters of methods, they are not mapped to this PSM
datetime	string	
String	string	

Table 11.5 - Data types for DCPS PSM

uint8	char	
uint16	short	
uint32	long	
uint64	long long	

11.1.4 Specific failure codes

N/A.

11.1.5 Conformance Criteria

This PSM acknowledges the same conformance criteria as the PIM (cf. Section 2). So, classes, attributes, and methods which are not known in a PIM compliance profile are not mapped to the corresponding PSM compliance profile. Moreover, this PSM knows two flavors:

- The DCPS/f ('/f' stands for full) relates to the full DCPS PSM as described in this section.

Issue 11517 - Root topics are not needed in PSM DCPS

- The DCPS/m ('/m' stands for monitoring) relates to the "element topics" and "indication topics" – i.e., all of them but "control topics." DCPS/m profile is therefore a subset of the DCPS/f profile.

Note – The DCPS/f is meant to be mandatory and the DCPS/m is meant to be optional. For example usable in combination with a CORBA/IDL PSM or a DMTF/CIM PSM.

Note – Static data have to be accessed via the mandatory XML PSM (cf. Chapter 9).

11.1.6 Mapping

Note – DDS Data types (cf. [DDS]) are defined through their equivalent CORBA/IDL structure, enumeration, and union.

Note – Key attributes are defined by keylist pragma. Authors are well aware that this is not a DDS standard construct (this is a Splice DDS facility) and will revise the specification when there is a standardized way of declaring keys.

Issue 11517 - Root topics are not needed in PSM DCPS

```
module AMSM {
// Indication topics
//
enum AMS_RTSoftwareStatus {
    SW_STARTED,
    SW_STOPPED,
    SW_FAILED
};
//
enum AMS_RTSWIndication_Case {
    AMS_ESE_IndicationCase,
```

```
    AMS_Application_IndicationCase,
    AMS_System_IndicationCase,
    AMS_LB_IndicationCase,
    AMS_FT_IndicationCase
} ;
//
enum AMS_StdHWUtilisation {
    HU_NONSTD,
    HU_CPU_LOAD,
    HU_NETWORK_LOAD,
    HU_NETWORK_BANDWIDTH,
    HU_VIRTUAL_MEMORY,
    HU_VIRTUAL_MEMORY_OCCUPATION,
    HU_TOTAL_MEMORY,
    HU_TOTAL_MEMORY_OCCUPATION,
    HU_PROCESS_NUMBER,
    HU_THREAD_NUMBER,
    HU_DISK,
    HU_DISK_OCCUPATION
};
//
union AMS_RTSWIndication_Element switch (AMS_RTSWIndication_Case) {
    case AMS_ESE_IndicationCase:
           // full name of elements of AMS_ExecutableSoftwareElement
           string ESE;
    case AMS_Application_IndicationCase:
           // full name of elements of AMS_Application
           string Application;
    case AMS_System_IndicationCase:
           // full name of elements of AMS_SoftwareSystem
           string System;
    case AMS_LB_IndicationCase:
           // full name of elements of AMS_LoadBalancingGroup
           string LB;
    case AMS_FT_IndicationCase:
           // full name of elements of AMS_RedundancyGroup
           string RG;
};
struct AMS_RTSWIndication {
    //
    AMS_RTSWIndication_Case ElementCase;
    AMS_RTSWIndication_Element Element;
    //
    AMS_RTSoftwareStatus AMS_RTSW;
    //
    string IndicationTime;
};
#pragma keylist AMS_RTSWIndication Element,AMS_RTSW

//
struct AMS_PropertyStdHWUtilisation {
    //
    string Name;
    AMS_StdHWUtilisation StdName;
    //
    string Value;
};
#pragma keylist AMS_PropertyStdHWUtilisation Name,StdName

//
```

```
struct AMS_RTHWIndication {
    // the full name of an element of AMS_ComputerSystem
    string AMS_NetworkElt;
    // AMS_Property<AMS_StdHWUtilisation, HU_NONSTD>
            sequence<AMS_PropertyStdHWUtilisation> AMS_RTHS;
    //
    string IndicationTime;
};
#pragma keylist AMS_RTHWIndication AMS_NetworkElt,CPULoad,MemoryLoad,DskUsage

// Control topics
    enum AMS_ErrorCode {
    AMS_UNKNOWN,
    AMS_BADFILTER,
    AMS_BADSUBSCRIPTIONID,
    AMS_BADCONNECTIVITY,
    AMS_BADDEVICES,
    AMS_BADRESOURCES,
    AMS_BADMODELTYPE,
    AMS_BADCOMMANDLINE,
    AMS_BADACTION,
    AMS_BADCHECK,
    AMS_BADSTATE,
    AMS_STARTFAILED,
    AMS_SHUTDOWNFAILED,
    AMS_LOADFAILED,
    AMS_STOPFAILED,
    AMS_CONTFAILED,
    AMS_DEPLOYFAILED,
    AMS_PRIMARYFAILED,
    AMS_RESOURCEERROR,
    AMS_RIGHTERROR,
    AMS_NOTCHECKED,
    AMS_ALREADYPRIMARY,
    AMS_NOTFT
    };

struct AMS_ErrorStruct {
    // identifier of the request
    long request_id;
    // error code
    AMS_ErrorCode Code;
    // message
    string Message;
    // errno
    long Number
    // full path to the element
    string Element
};
#pragma keylist AMS_Response request_id

//
struct AMS_CT_LoadConfiguration {
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
    string file;
};
```

```
#pragma keylist AMS_CT_LoadConfiguration Element

//
struct AMS_CT_UnloadConfiguration {
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
    string file;
};
#pragma keylist AMS_CT_UnloadConfiguration Element

//
struct AMS_CT_ShutDownESE {
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
    string filter;
    sequance<AMS_State> stateFilters;
};
#pragma keylist AMS_CT_ShutDownESE Element

//
struct AMS_CT_CreateHardwareGroup {
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
    string location;
    string connectivity;
    string devices;
};
#pragma keylist AMS_CT_CreateHardwareGroup Element

//
struct AMS_CT_ShutDown {
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
};
#pragma keylist AMS_CT_ShutDown Element

//
struct AMS_CT_StartUp {
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
};
#pragma keylist AMS_CT_StartUp Element

//
struct AMS_CT_StartUpOnHost {
```

```
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
    string host;
    sring commandLine;
};
#pragma keylist AMS_CT_StartUpOnHost Element

//
struct AMS_CT_StartUpOnSpec {
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
    string spec;
};
#pragma keylist AMS_CT_StartUpOnSpec Element

//
struct AMS_CT_ActivateAsPrimary {
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
};
#pragma keylist AMS_CT_ActivateAsPrimary Element

//
struct AMS_CT_Continue {
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
};
#pragma keylist AMS_CT_Continue Element

//
struct AMS_CT_ForceShutDown {
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
};
#pragma keylist AMS_CT_ForceShutDown Element

//
struct AMS_CT_Load {
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
};
#pragma keylist AMS_CT_Load Element
```

```
//
struct AMS_CT_Stop {
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
};
#pragma keylist AMS_CT_Stop Element

//
struct AMS_CT_ClearLog {
    // identifier of the request
    long request_id;
    // the element to be controled
    string Element;
    // the attributes (cf. PIM)
};
#pragma keylist AMS_CT_ClearLog Element

// package 'Supported Application Model'
//
enum AMS_OSType {
    Unknown,
    Other,
    MACOS,
    ATTUNIX,
    DGUX,
    DECNT,
    Tru64UNIX,
    OpenVMS,
    HPUX,
    AIX,
    MVS,
    OS400,
    OS_2,
    JavaVM,
    MSDOS,
    WIN3x,
    WIN95,
    WIN98,
    WINNT,
    WINCE,
    NCR3000,
    NetWare,
    OSF,
    DC_OS,
    ReliantUNIX,
    SCOUnixWare,
    SCOOpenServer,
    Sequent,
    IRIX,
    Solaris,
    SunOS,
    U6000,
    ASERIES,
    TandemNSK,
    TandemNT,
    BS2000,
```

```
        LINUX,
        Lynx,
        XENIX,
        VM,
        InteractiveUNIX,
        BSDUNIX,
        FreeBSD,
        NetBSD,
        GNUHurd,
        OS9,
        MACHKernel,
        Inferno,
        QNX,
        EPOC,
        IxWorks,
        VxWorks,
        MiNT,
        BeOS,
        HPMPE,
        NextStep,
        PalmPilot,
        Rhapsody,
        Windows2000,
        Dedicated,
        OS_390,
        VSE,
        TPF,
        Windows_R_Me,
        CalderaOpenUNIX,
        OpenBSD,
        NotApplicable,
        WindowsXP,
        z_OS,
```

Issue 11519 - Vista

```
        WindowsVista
};
//
enum AMS_ModelType {
        AMS_PROCESS,
        AMS_J2EE,
        AMS_CCM
};
//
enum AMS_Control {
        AMS_LOAD,
        AMS_LOAD_START,
        AMS_START,
        AMS_STOP,
        AMS_HALT,
        AMS_CONTINUE,
        AMS_SHUTDOWN,
        AMS_RECOVER,
        AMS_UNLOAD,
        AMS_LOAD_DIRTY,
        LOAD_START_DIRTY,
        AMS_STOP_HALTED,
        AMS_RECLAIM,
        AMS_ALLOCATE,
        AMS_RECOVER_DIRTY
```

```
};
//
enum AMS_State {
    AMS_EXECUTABLE,
    AMS_HALTED,
    AMS_LOADED,
    AMS_RUNNING,
    AMS_STOPPED,
    AMS_UNALLOCATED,
    AMS_ERROR
};
//
enum AMS_StdMechanisms {
    MS_NONSTD,
    MS_POSIXSIGNAL
};
//
struct AMS_ControlMechanismCouple {
    //
    AMS_Control Control;
    //
    AMS_StdMechanism StdName;
    string Name;
};
#pragma keylist AMS_ControlMechanismCouple Control,Name,StdName

//
struct AMS_SupportedApplicationModel {
    //
    sequence<AMS_ModelType> SupportedModelType;
    //
    sequence<AMS_OSType> SupportedOSType;
    //
    sequence<AMS_Control> SupportedControls;
    //
    sequence<AMS_State> SupportedStates;
    //
    string Name;
    //
    string ConfigurationInfo;
    //
    sequence<AMS_ControlMechanismCouple> SupportedMechanisms;
};
#pragma keylist AMS_SupportedApplicationModel Name



// package 'Application'
//
enum AMS_ReplicationStyle {
    RG_COLD_PASSIVE,
    RG_WARM_PASSIVE,
    RG_ACTIVE,
    RG_ACTIVE_WITH_VOTING,
    RG_STATELESS,
    RG_IMPL_DEFINED
};
//
enum AMS_BalancingStyle {
    LB_ROUND_ROBIN,
```

```
    LB_RANDOM,
    LB_IMPL_DEFINED
};
//
enum AMS_RedundancyEltState {
    REDSTATE_NORG,
    REDSTATE_PRIMARY,
    REDSTATE_PASSIVE
};
//
enum AMS_StdState {
    ST_NONSTD,
    ST_ENV
};
//
struct AMS_SoftwareSystem {
    // the full name of an element of AMS_SoftwareSystem
    sequence<string, 1> Owner
    // sequence of full names of elements of AMS_Application
    // sequence of full names of elements of AMS_ExecutableSoftwareElement
    sequence<string> CIM_SystemComponent;
    // sequence of full names of elements of AMS_SoftwareSystem
    sequence<string> AMS_SystemOfSystem;
    // the full name of an element of AMS_SoftwareFeatureSpec
    string AMS_SystemFeature;
    //
    AMS_RTSoftwareStatus Status;
    //
    string Name;
};
#pragma keylist AMS_SoftwareSystem Name,Owner

//
struct AMS_Application {
    // the full name of an element of AMS_SoftwareSystem
    // the full name of an element of AMS_Application
    string Owner
    // sequence of full names of elements of AMS_ExecutableSoftwareElement
    // sequence of full names of elements of AMS_RedundancyGroup
    // sequence of full names of elements of AMS_LoadBalancingGroup
    sequence<string> CIM_SystemComponent;
    // the full name of an element of AMS_SoftwareFeatureSpec
    string AMS_ApplicationFeature;
    // sequence of full names of elements of AMS_Application
    sequence<string> AMS_ApplicationOfApplication;
    //
    AMS_RTSoftwareStatus Status;
    //
    string Name;
};
#pragma keylist AMS_Application Name,Owner

//
struct AMS_PropertyStdState {
    //
    string Name;
    AMS_StdState StdName;
    //
    string Value;
};
```

```
#pragma keylist AMS_PropertyStdState Name,StdName

//
struct AMS_ExecutableSoftwareElement {
    // the full name of an element of AMS_SoftwareSystem
    // the full name of an element of AMS_Application
    string Owner
    // sequence of full names of elements of AMS_Application
    sequence<string> CIM_SystemComponent;
    // the full name of an element of AMS_ESESpec
    string CIM_SoftwareElementServiceImplementation;
    //
    AMS_State CurrentState;
    // sequence of full names of elements of AMS_RedundancyGroup
    // sequence of full names of elements of AMS_LoadBalancingGroup
    sequence<string, 1> CIM_RedundancyComponent;
    //
    AMS_RTSoftwareStatus Status;
    //
    AMS_RedundancyEltState RedState;
    // the full name of an element of AMS_DeploymentLink
    string AMS_ESEDeployed;
    //
    string Name;
    // platform-specific states
    sequence<AMS_PropertyStdState> AMS_SpecificStates;
    // sequence of full names of CIM_Process
    sequence<string> CIM_ServiceProcess;
};
#pragma keylist AMS_ExecutableSoftwareElement Name,Owner

//
struct AMS_RedundancyGroup {
    // the full name of an element of AMS_Application
    string Owner
    // sequence of full names of elements of AMS_ExecutableSoftwareElement
    sequence<string> CIM_RedundancyComponent;
    // the full name of an element of AMS_SoftwareFeatureSpec
    string AMS_RedundancyFeature;
    //
    AMS_ReplicationStyle Style;
    //
    AMS_RTSoftwareStatus Status;
    //
    string Name;
};
#pragma keylist AMS_RedundancyGroup Name,Owner

//
struct AMS_LoadBalancingGroup {
    // the full name of an element of AMS_Application
    string Owner
    // the full name of an element of AMS_SoftwareFeatureSpec
    string AMS_LoadBalancingFeature;
    // sequence of full names of elements of AMS_ExecutableSoftwareElement
    sequence<string> CIM_RedundancyComponent;
    //
    AMS_BalancingStyle Style;
    //
    AMS_RTSoftwareStatus Status;
```

```
    //
    string Name;
};
#pragma keylist AMS_LoadBalancingGroup Name,Owner



// package 'Application Deployment'
//
struct AMS_DeploymentLink {
    // the full name of an element of AMS_DeploymentConfiguration
    string Owner
    // the full name of an element of AMS_Host
    string AMS_HostUsed;
    // the full name of an element of AMS_ExecutableSoftwareElement
    string AMS_ESEDeployed;
    //
    string LinkID;
};
#pragma keylist AMS_DeploymentLink LinkID,Owner

//
struct AMS_DeploymentConfiguration {
    // sequence of full names of elements of AMS_DeploymentLink
    sequence<string> AMS_DeploymentLinks;
    // the full name of an element of AMS_DeploymentSpec
    string AMS_DeploymentSpecAssoc;
    //
    string Name;
};
#pragma keylist AMS_DeploymentConfiguration Name



// package 'CIM'
//
struct CIM_Process {
    //
    string Name;
    //
    string Handle;
    // sequence of full names of elements of AMS_ExecutbaleSoftwareElement
    list<string> CIM_ServiceProcess;
    // sequence of full names of elements of CIM_Thread
    list<string> CIM_ProcessThead;
};
#pragma keylist CIM_Process Handle

//
struct CIM_UnixProcess {
    //
    string Name;
    //
    string Handle;
    // sequence of full names of elements of AMS_ExecutbaleSoftwareElement
    list<string> CIM_ServiceProcess;
    // sequence of full names of elements of CIM_Thread
    list<string> CIM_ProcessThead;
    //
    long long ProcessGroupID;
```

```
    //
    long long RealUserID;
    //
    list<string> Parameters;
};
#pragma keylist CIM_UnixProcess Handle

//
struct CIM_Thread {
    //
    string Name;
    //
    string Handle;
    //
    long Priority;
    // the full name of an element of CIM_Process
    // the full name of an element of CIM_UnixProcess
    string Owner;
};
#pragma keylist CIM_Thread Handle,Owner

//
struct CIM_IPProtocolEndPoint {
    // the full name of an element of CIM_ComputerSystem
    // the full name of an element of CIM_ConnectivityCollection
    string Owner
    // sequence of full names of elements of CIM_ProtocolEndPoint
    // sequence of full names of elements of CIM_ProtocolEndPoint
    sequence<string> CIM_EndpointIdentity;
    // sequence of full names of elements of CIM_NextHopRoute
    sequence<string> CIM_RouteUsesEndpoint;
    // sequence of full names of elements of CIM_LANEndPoint
    sequence<string> CIM_BindsToLANEndPoint;
    //
    string Name;
    //
    string IPv4Address;
    //
    string IPv6Address;
    //
    string SubnetMask;
    //
    char PrefixLength;
};
#pragma keylist CIM_IPProtocolEndPoint Name,Owner

//
struct CIM_NextHopRoute {
    // the full name of an element of AMS_Host
    // the full name of an element of AMS_Router
    string Owner
    // sequence of full names of elements of CIM_RemoteServiceAccessPoint
    sequence<string, 1> CIM_AssociatedNextHop;
    // sequence of full names of elements of CIM_ProtocolEndPoint
    sequence<string, 1> CIM_RouteUsesEndpoint;
    //
    string InstanceID;
    //
    string DestinationAddress;
    //
```

```
    short AdminDistance;
    //
    short RouteMetric;
    //
    boolean IsStatic;
    //
    short TypeOfRoute;
};
#pragma keylist CIM_NextHopRoute InstanceID,Owner

//
struct CIM_Location {
    // sequence of full names of elements of CIM_PhysicalElement
    sequence<string> CIM_PhysicalElementLocation;
    // sequence of full names of elements of AMS_ComputerSystem
    sequence<string> CIM_ElementLocation;
    //
    string Name;
};
#pragma keylist CIM_Location Name

//
struct CIM_ProtocolEndPoint {
    // the full name of an element of CIM_ComputerSystem
    // the full name of an element of CIM_ConnectivityCollection
    string Owner
    // sequence of full names of elements of CIM_ProtocolEndPoint
    // sequence of full names of elements of CIM_ProtocolEndPoint
    sequence<string> CIM_EndpointIdentity;
    // sequence of full names of elements of CIM_NextHopRoute
    sequence<string> CIM_RouteUsesEndpoint;
    // sequence of full names of elements of CIM_LANEndPoint
    sequence<string> CIM_BindsToLANEndPoint;
    //
    string Name;
};
#pragma keylist CIM_ProtocolEndPoint Name,Owner

//
struct CIM_LogicalDisk {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of CIM_PhysicalElement
    sequence<string> CIM_Realizes;
    //
    string DeviceID;
};
#pragma keylist CIM_LogicalDisk DeviceID,Owner

//
struct CIM_Memory {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of CIM_PhysicalElement
    sequence<string> CIM_Realizes;
    //
    string DeviceID;
};
#pragma keylist CIM_Memory DeviceID,Owner
```

```
//
struct CIM_PowerSupply {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of CIM_PhysicalElement
    sequence<string> CIM_Realizes;
    //
    string DeviceID;
};
#pragma keylist CIM_PowerSupply DeviceID,Owner

//
struct CIM_Watchdog {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of CIM_PhysicalElement
    sequence<string> CIM_Realizes;
    //
    string DeviceID;
};
#pragma keylist CIM_Watchdog DeviceID,Owner

//
struct CIM_Processor {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of CIM_PhysicalElement
    sequence<string> CIM_Realizes;
    //
    string DeviceID;
    //
    short LoadPercentage;
    //
    short CPUStatus;
};
#pragma keylist CIM_Processor DeviceID,Owner

//
struct CIM_LogicalPort {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of CIM_PhysicalElement
    sequence<string> CIM_Realizes;
    //
    string DeviceID;
};
#pragma keylist CIM_LogicalPort DeviceID,Owner

//
struct CIM_NetworkPort {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of CIM_PhysicalElement
    sequence<string> CIM_Realizes;
    //
    string DeviceID;
};
#pragma keylist CIM_NetworkPort DeviceID,Owner

//
```

```
struct CIM_EthernetPort {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of CIM_PhysicalElement
    sequence<string> CIM_Realizes;
    //
    string DeviceID;
};
#pragma keylist CIM_EthernetPort DeviceID,Owner

//
struct CIM_Display {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of CIM_PhysicalElement
    sequence<string> CIM_Realizes;
    //
    string DeviceID;
};
#pragma keylist CIM_Display DeviceID,Owner

//
struct CIM_Sensor {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of CIM_PhysicalElement
    sequence<string> CIM_Realizes;
    //
    string DeviceID;
};
#pragma keylist CIM_Sensor DeviceID,Owner

//
struct CIM_StorageExtent {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of CIM_PhysicalElement
    sequence<string> CIM_Realizes;
    //
    string DeviceID;
};
#pragma keylist CIM_StorageExtent DeviceID,Owner

//
struct CIM_ConnectivityCollection {
    // the full name of an element of CIM_Network
    string Owner
    // sequence of full names of elements of CIM_ProtocolEndPoint
    sequence<string> CIM_MemberOfCollection;
    //
    long long InstanceID;
};
#pragma keylist CIM_ConnectivityCollection InstanceID,Owner

//
struct CIM_EthernetPortStatistics {
    // the full name of an element of CIM_LanEndPoint
    string Owner
    //
    long long InstanceID;
```

```
    //
    long long BytesTransmitted;
    //
    long long BytesReceived;
    //
    long long PacketsTransmitted;
    //
    long long PacketsReceived;
};
#pragma keylist CIM_EthernetPortStatistics InstanceID,Owner

//
struct CIM_LANEndPoint {
    // the full name of an element of AMS_ComputerSystem
    // the full name of an element of CIM_ConnectivityCollection
    string Owner
    // sequence of full names of elements of CIM_EthernetPortStatistics
    sequence<string> CIM_ElementStatisticalData;
    // sequence of full names of elements of CIM_ProtocolEndPoint
    // sequence of full names of elements of CIM_ProtocolEndPoint
    sequence<string> CIM_EndpointIdentity;
    // sequence of full names of elements of CIM_NextHopRoute
    sequence<string> CIM_RouteUsesEndpoint;
    // sequence of full names of elements of CIM_ServiceAccessPoint
    // sequence of full names of elements of CIM_LANEndPoint
    sequence<string> CIM_BindsToLANEndPoint;
    //
    string Name;
    //
    string LANID;
    //
    string MACAddress;
    //
    string AliasAddresses;
    //
    string GroupAddresses;
    //
    long MaxDataSize;
};
#pragma keylist CIM_LANEndPoint Name,Owner

//
struct CIM_Network {
    // sequence of full names of elements of CIM_ConnectivityCollection
    sequence<string> CIM_HostedCollection;
    // sequence of full names of elements of CIM_Network
    sequence<string> AMS_SubnetComponent;
    //
    string Name;
};
#pragma keylist CIM_Network Name

//
struct CIM_NextHopIPRoute {
    // the full name of an element of AMS_ComputerSystem
    // the full name of an element of CIM_ConnectivityCollection
    string Owner
    // sequence of full names of elements of CIM_RemoteServiceAccessPoint
    sequence<string, 1> CIM_AssociatedNextHop;
    // sequence of full names of elements of CIM_ProtocolEndPoint
```

```
    sequence<string, 1> CIM_RouteUsesEndpoint;
    //
    string InstanceID;
    //
    string DestinationAddress;
    //
    short AdminDistance;
    //
    short RouteMetric;
    //
    boolean IsStatic;
    //
    short TypeOfRoute;
    //
    short RouteDerivation;
    //
    string DestinationMask;
    //
    char PrefixLength;
    //
    short AddressType;
};
#pragma keylist CIM_NextHopIPRoute InstanceID,Owner

//
struct CIM_RemoteServiceAccessPoint {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of CIM_NextHopRoute
    sequence<string> CIM_AssociatedNextHop;
    // sequence of full names of elements of CIM_LANEndPoint
    sequence<string> CIM_BindsToLANEndPoint;
    //
    string Name;
    //
    string AccessInfo;
    //
    short InfoFormat;
};
#pragma keylist CIM_RemoteServiceAccessPoint Name,Owner

//
struct CIM_ServiceAccessPoint {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of CIM_LANEndPoint
    sequence<string> CIM_BindsToLANEndPoint;
    //
    string Name;
};
#pragma keylist CIM_ServiceAccessPoint Name,Owner



// package 'Logical Hardware'
//
struct AMS_Host {
    // sequence of full names of elements of CIM_ServiceAccessPoint
    sequence<string> CIM_HostedAccessPoint;
    // sequence of full names of elements of CIM_NextHopRoute
```

```
    sequence<string> CIM_HostedRoute;
    // sequence of full names of elements of CIM_LogicalDevice
    // sequence of full names of elements of AMS_Domain
    // sequence of full names of elements of AMS_HardwareGroup
    sequence<string> CIM_SystemComponent;
    // sequence of full names of elements of CIM_Location
    sequence<string, 1> CIM_ElementLocation;
    // sequence of full names of elements of AMS_OperatingSystem
    sequence<string> CIM_InstalledOS;
    // sequence of full names of elements of AMS_OperatingSystem
    sequence<string> CIM_RunningOS;
    // AMS_Property<AMS_StdHWUtilisation, HU_NONSTD>
           sequence<AMS_PropertyStdHWUtilisation> AMS_RTHU;
    // sequence of full names of elements of AMS_ConfigurationSpecification
    sequence<string, 1> AMS_ConfSpecCS;
    // sequence of full names of elements of AMS_DeploymentLink
    sequence<string> AMS_HostUsed;
    //
    string Name;
    //
    string ArchitectureInfo;
    //
    short Status;
    //
    short NetworkLoad;
};
#pragma keylist AMS_Host Name

//
struct AMS_Router {
    // sequence of full names of elements of CIM_ServiceAccessPoint
    sequence<string> CIM_HostedAccessPoint;
    // sequence of full names of elements of CIM_LogicalDevice
    // sequence of full names of elements of AMS_Domain
    // sequence of full names of elements of AMS_HardwareGroup
    sequence<string> CIM_SystemComponent;
    // sequence of full names of elements of CIM_Location
    sequence<string, 1> CIM_ElementLocation;
    // sequence of full names of elements of AMS_OperatingSystem
    sequence<string> CIM_InstalledOS;
    // sequence of full names of elements of AMS_OperatingSystem
    sequence<string> CIM_RunningOS;
    // sequence of full names of elements of CIM_NextHopRoute
    sequence<string> CIM_HostedRoute;
    // sequence of full names of elements of AMS_ConfigurationSpecification
    sequence<string, 1> AMS_ConfSpecCS;
    //
    string Name;
    //
    string ArchitectureInfo;
    //
    short Status;
    //
    short NetworkLoad;
};
#pragma keylist AMS_Router Name

//
struct AMS_Switch {
    // sequence of full names of elements of CIM_ServiceAccessPoint
```

```
    sequence<string> CIM_HostedAccessPoint;
    // sequence of full names of elements of CIM_LogicalDevice
    // sequence of full names of elements of AMS_Domain
    // sequence of full names of elements of AMS_HardwareGroup
    sequence<string> CIM_SystemComponent;
    // sequence of full names of elements of CIM_Location
    sequence<string, 1> CIM_ElementLocation;
    // sequence of full names of elements of AMS_OperatingSystem
    sequence<string> CIM_InstalledOS;
    // sequence of full names of elements of AMS_OperatingSystem
    sequence<string> CIM_RunningOS;
    // sequence of full names of elements of AMS_ConfigurationSpecification
    sequence<string, 1> AMS_ConfSpecCS;
    //
    string Name;
    //
    string ArchitectureInfo;
    //
    short Status;
    //
    short NetworkLoad;
};
#pragma keylist AMS_Switch Name

//
struct AMS_HardwareGroup {
    // sequence of full names of elements of AMS_ComputerSystem
    sequence<string> CIM_SystemComponent;
    // sequence of full names of elements of AMS_ConfigurationSpecification
    sequence<string, 1> AMS_ConfSpecHG;
    //
    string Name;
};
#pragma keylist AMS_HardwareGroup Name

//
struct AMS_Printer {
    // sequence of full names of elements of CIM_ServiceAccessPoint
    sequence<string> CIM_HostedAccessPoint;
    // sequence of full names of elements of CIM_LogicalDevice
    // sequence of full names of elements of AMS_Domain
    // sequence of full names of elements of AMS_HardwareGroup
    sequence<string> CIM_SystemComponent;
    // sequence of full names of elements of CIM_Location
    sequence<string, 1> CIM_ElementLocation;
    // sequence of full names of elements of AMS_OperatingSystem
    sequence<string> CIM_InstalledOS;
    // sequence of full names of elements of AMS_OperatingSystem
    sequence<string> CIM_RunningOS;
    // sequence of full names of elements of AMS_ConfigurationSpecification
    sequence<string, 1> AMS_ConfSpecCS;
    //
    string Name;
    //
    string ArchitectureInfo;
    //
    short Status;
    //
    short NetworkLoad;
};
```

```
#pragma keylist AMS_Printer Name

//
struct AMS_DomainManager {
    // sequence of full names of elements of CIM_ServiceAccessPoint
    sequence<string> CIM_HostedAccessPoint;
    // sequence of full names of elements of AMS_Domain
    sequence<string, 1> AMS_DomainManagerRole;
    // sequence of full names of elements of CIM_NextHopRoute
    sequence<string> CIM_HostedRoute;
    // sequence of full names of elements of CIM_LogicalDevice
    // sequence of full names of elements of AMS_Domain
    // sequence of full names of elements of AMS_HardwareGroup
    sequence<string> CIM_SystemComponent;
    // sequence of full names of elements of CIM_Location
    sequence<string, 1> CIM_ElementLocation;
    // sequence of full names of elements of AMS_OperatingSystem
    sequence<string> CIM_InstalledOS;
    // sequence of full names of elements of AMS_OperatingSystem
    sequence<string> CIM_RunningOS;
    // AMS_Property<AMS_StdHWUtilisation, HU_NONSTD>
           sequence<AMS_PropertyStdHWUtilisation> AMS_RTHU;
    // sequence of full names of elements of AMS_ConfigurationSpecification
    sequence<string, 1> AMS_ConfSpecCS;
    // sequence of full names of elements of AMS_DeploymentLink
    sequence<string> AMS_HostUsed;
    //
    string Name;
    //
    string ArchitectureInfo;
    //
    short Status;
    //
    short NetworkLoad;
};
#pragma keylist AMS_DomainManager Name

//
struct AMS_Domain {
    // sequence of full names of elements of AMS_DomainManager
    sequence<string> AMS_DomainManagerRole;
    // sequence of full names of elements of AMS_ComputerSystem
    sequence<string> CIM_SystemComponent;
    // sequence of full names of elements of AMS_ConfigurationSpecification
    sequence<string, 1> AMS_ConfSpecDom;
    //
    string Name;
};
#pragma keylist AMS_Domain Name

//
struct AMS_OperatingSystem {
    // the full name of an element of AMS_ComputerSystem
    string Owner
    // sequence of full names of elements of AMS_SupportedApplicationModel
    sequence<string> AMS_AMSupportedByOS;
    //
    AMS_OSType OSType;
    // sequence of full names of elements of AMS_ConfigurationSpecification
    sequence<string, 1> AMS_ConfSpecOS;
```

```
    //
    string Name;
    //
    string Version;
};
#pragma keylist AMS_OperatingSystem Name,Owner

//
struct AMS_LANEndPoint {
    // the full name of an element of AMS_ComputerSystem
    // the full name of an element of CIM_ConnectivityCollection
    string Owner
    // sequence of full names of elements of CIM_EthernetPortStatistics
    sequence<string> CIM_ElementStatisticalData;
    // sequence of full names of elements of CIM_ProtocolEndPoint
    // sequence of full names of elements of CIM_ProtocolEndPoint
    sequence<string> CIM_EndpointIdentity;
    // sequence of full names of elements of CIM_NextHopRoute
    sequence<string> CIM_RouteUsesEndpoint;
    // sequence of full names of elements of CIM_ServiceAccessPoint
    // sequence of full names of elements of CIM_LANEndPoint
    sequence<string> CIM_BindsToLANEndPoint;
    //
    string Name;
    //
    string LANID;
    //
    string MACAddress;
    //
    string AliasAddresses;
    //
    string GroupAddresses;
    //
    long MaxDataSize;
    //
    short Status;
    //
    short NetworkLoad;
};
#pragma keylist AMS_LANEndPoint Name,Owner



};
```

12 XML for HPI Platform Specific Model

12.1 Mapping Rationale

The objective of this PSM is to define the syntax and semantics of XML files which provide the minimal information about the system physical hardware, as needed by any compliant implementation.

This includes in particular the mapping between “logical hardware” (defined in Section 7.1.10) and “physical hardware.” An implementation is not required to provide an interface to physical hardware, for configuration, control and monitoring purposes, such as the SA-Forum Hardware Platform Interface (described in Section 6.1.2).

Mapping between “logical hardware” and “physical hardware” is needed to allow the identification of a faulty physical element, to be replaced and/or repaired (during system operation, or as part of maintenance).

In the CIM model this standard is based upon, an hardware physical element is represented through an instance of CIM_LogicalDevice, which is associated through the CIM_Realizes association (cf. Section 7.1.8.22) with one or more CIM_PhysicalElement. Conversely, a CIM_PhysicalElement may be associated with more than one CIM_LogicalDevice.

The XML platform specific model explicates the association between physical elements (PhysicalEntity) and logical elements (LogicalEntity). Physical elements are named according to the HPI naming convention for hardware entities (a “pathname” reflects the physical hierarchical structure), and logical elements are named according to whatever appropriate naming convention.

A LogicalEntity may be associated to one PhysicalEntity (in which case, this is a pure “renaming”). In addition, LogicalEntities may be associated through 2 intermediate elements:

EntityGroup associates one LogicalEntity L with several LogicalEntities, meaning that L represents a group of logical entities (which may typically be directly associated with physical entities).

EntityPartition associates a set of Logical Entities L1..Ln with another Logical Entity (which may typically be directly associated with a physical entity), meaning that L1..Ln represent a partition of the other entity.

The example below shows a combination of groups and partitions.

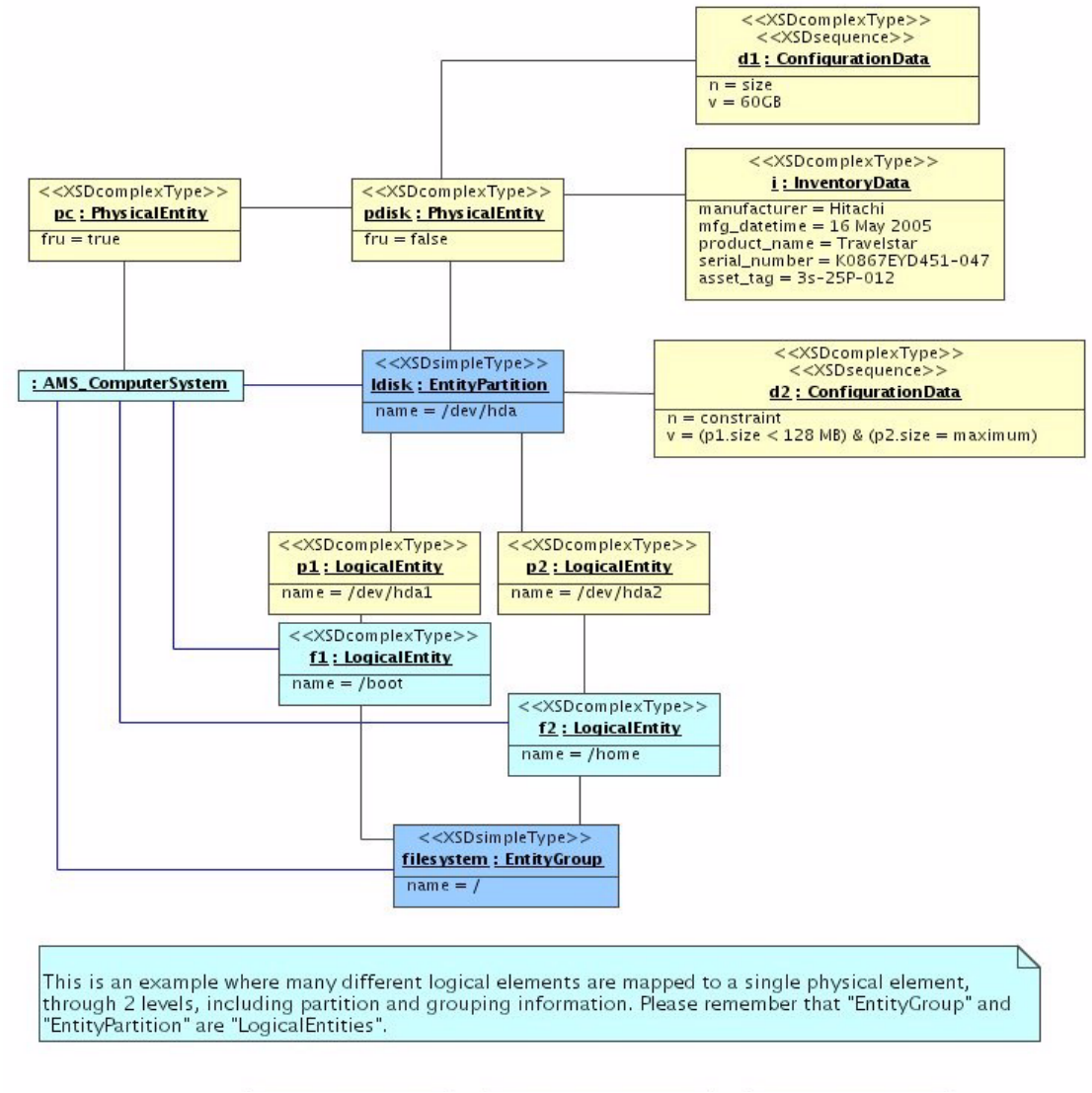


Figure 12.1 - Example of data in the “XML for HPI” PSM

The PSM is structured as shown by the diagram below:

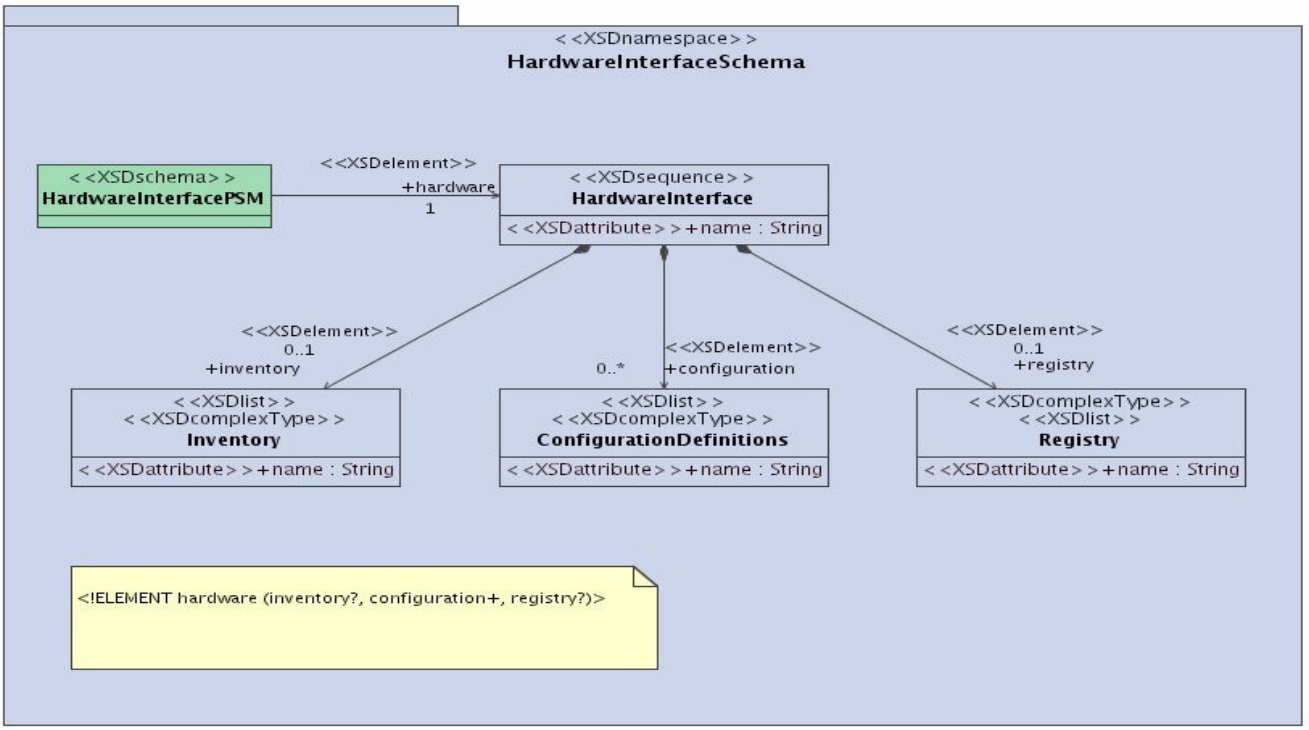


Figure 12.2 - Overall structure of the "XML for HPI" PSM

The CIM_Realizes association is made explicit, as explained above, in the "ConfigurationDefinitions" section.

In addition:

- The optional "Inventory" section provides detailed (static) inventory information about physical elements.
- The optional "Registry" section provides the current system configuration, possibly including (dynamic) status information.

The diagram on the next page (not part of the standard) illustrates how the XML PSM may be used as a public interface to the hardware and low-level software configuration management – and in particular in relation with an HPI (or IPMI) compliant implementation.

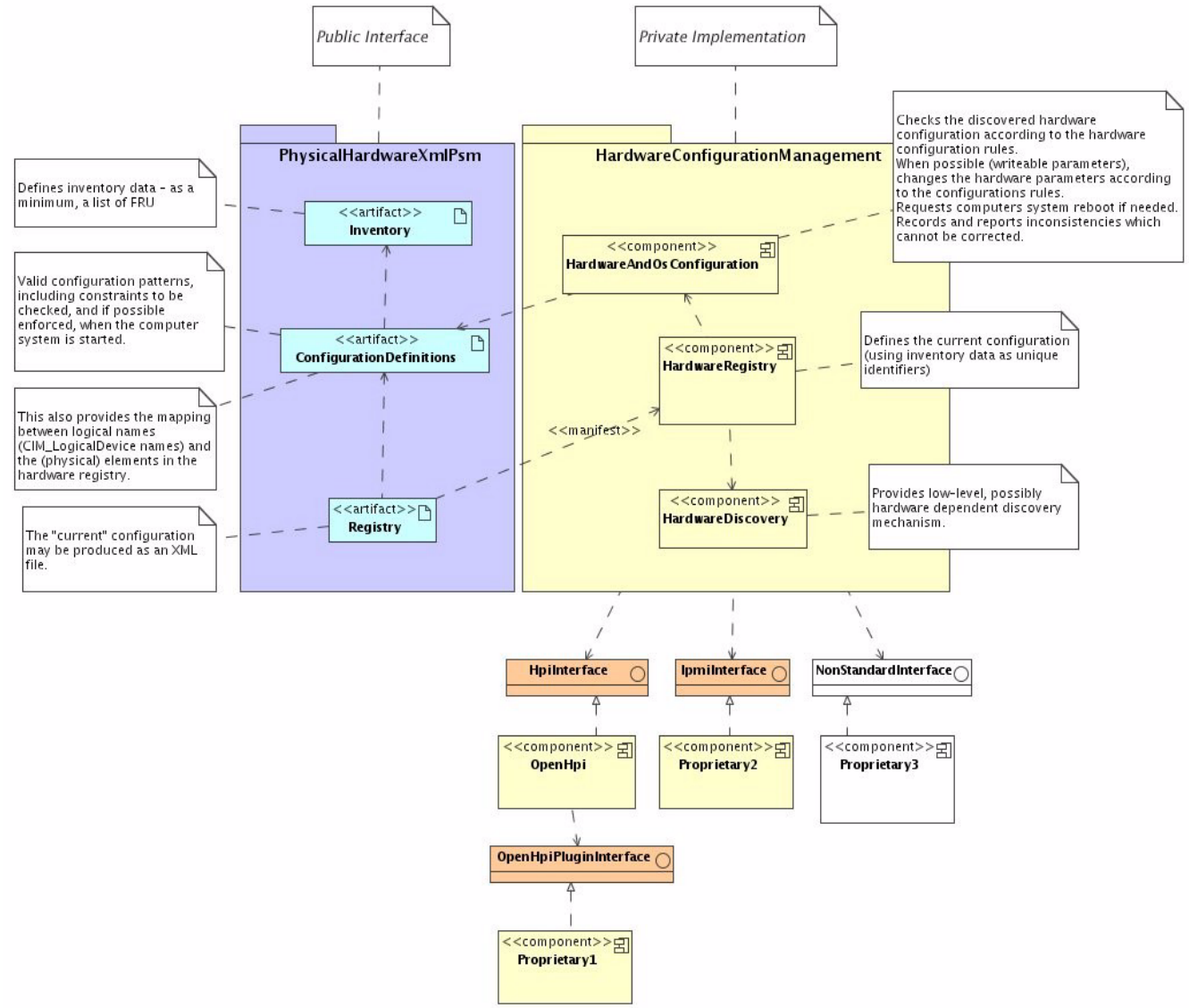


Figure 12.3 - Example of an implementation of the "XML for HPI" PSM

The 3 sections of the XML PSM are detailed by the 3 diagrams below.

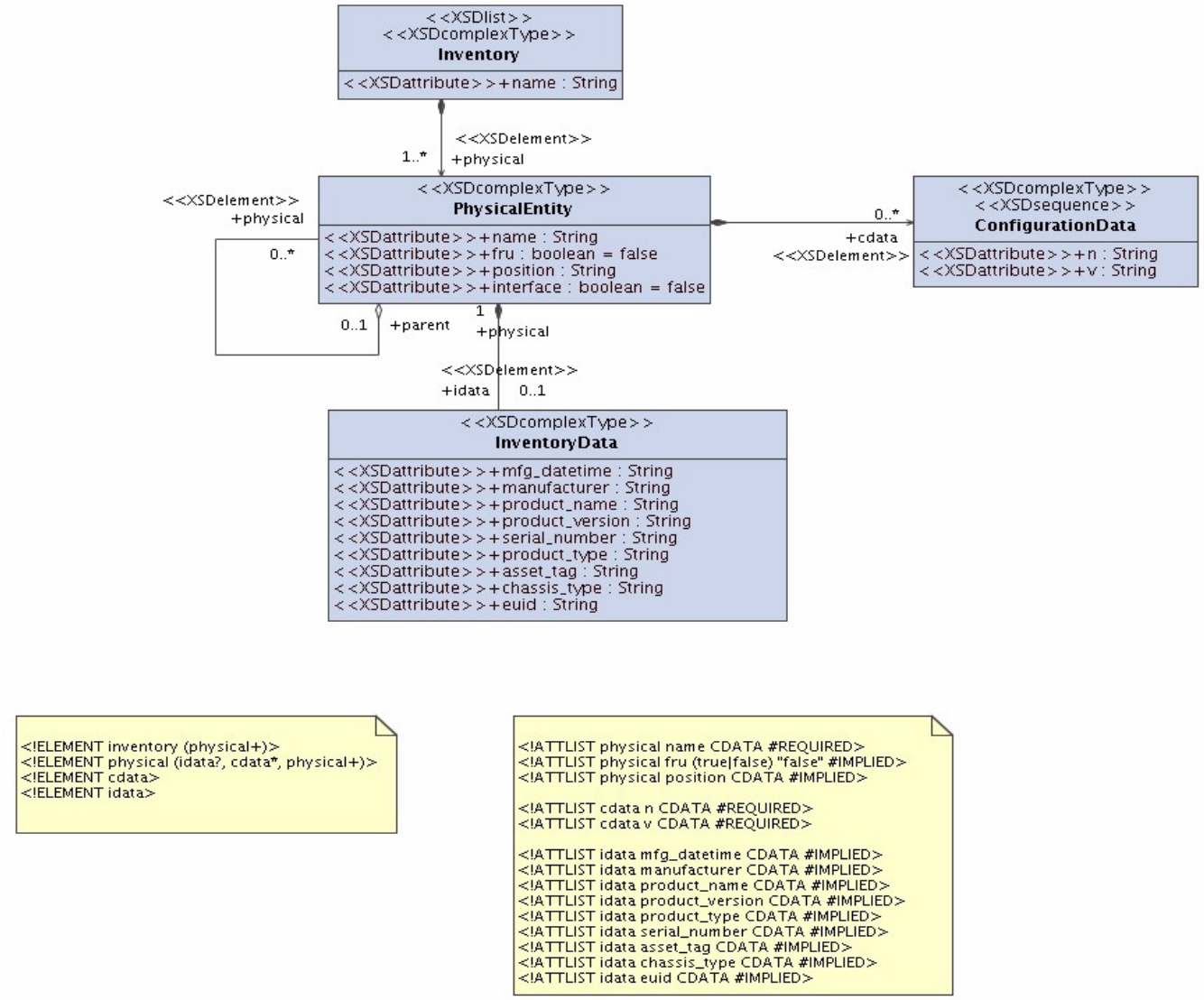


Figure 12.4 - "Inventory" part of the "XML for HPI" PSM

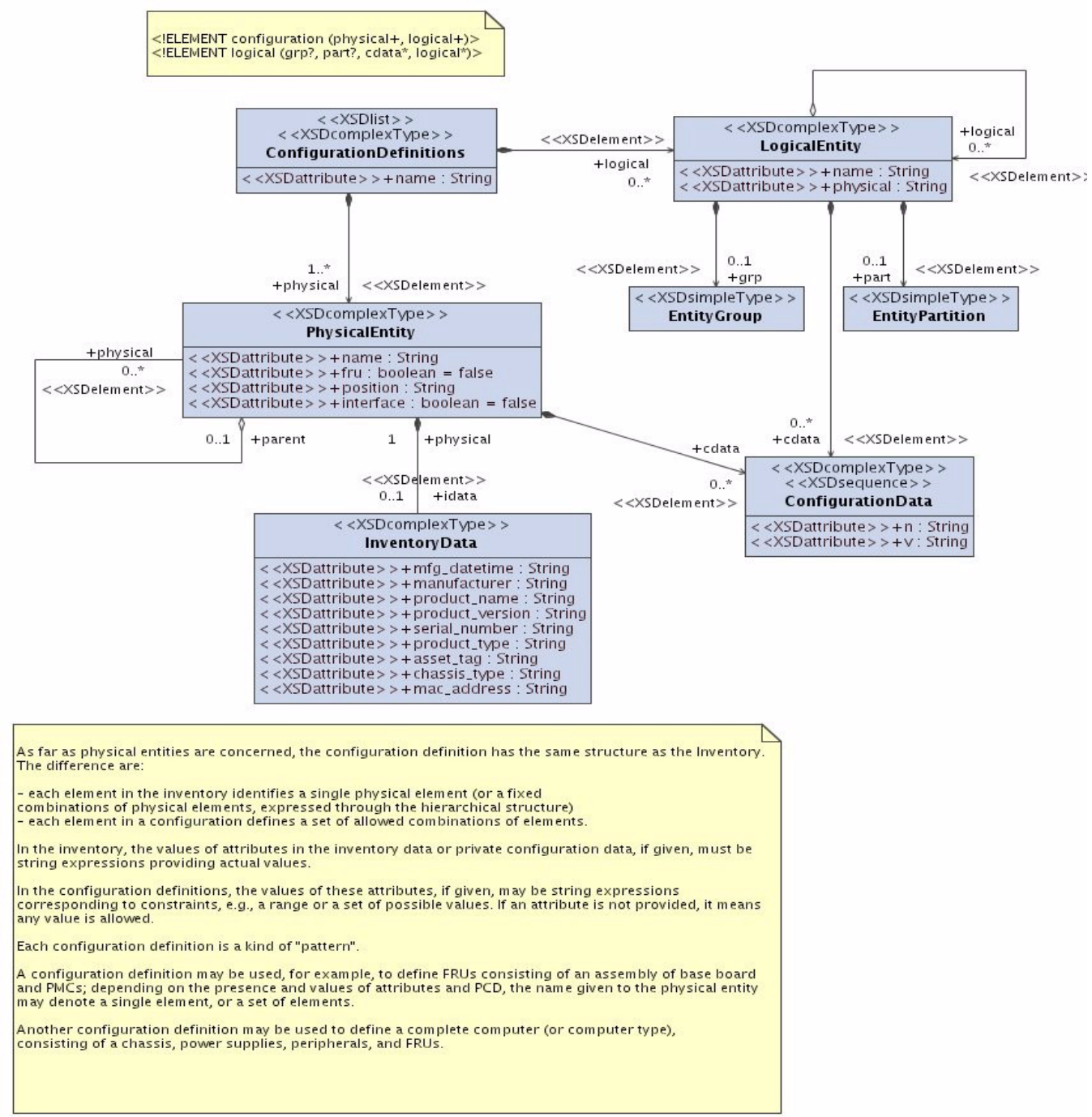


Figure 12.5 - "Configuration Definitions" part of the "XML for HPI" PSM

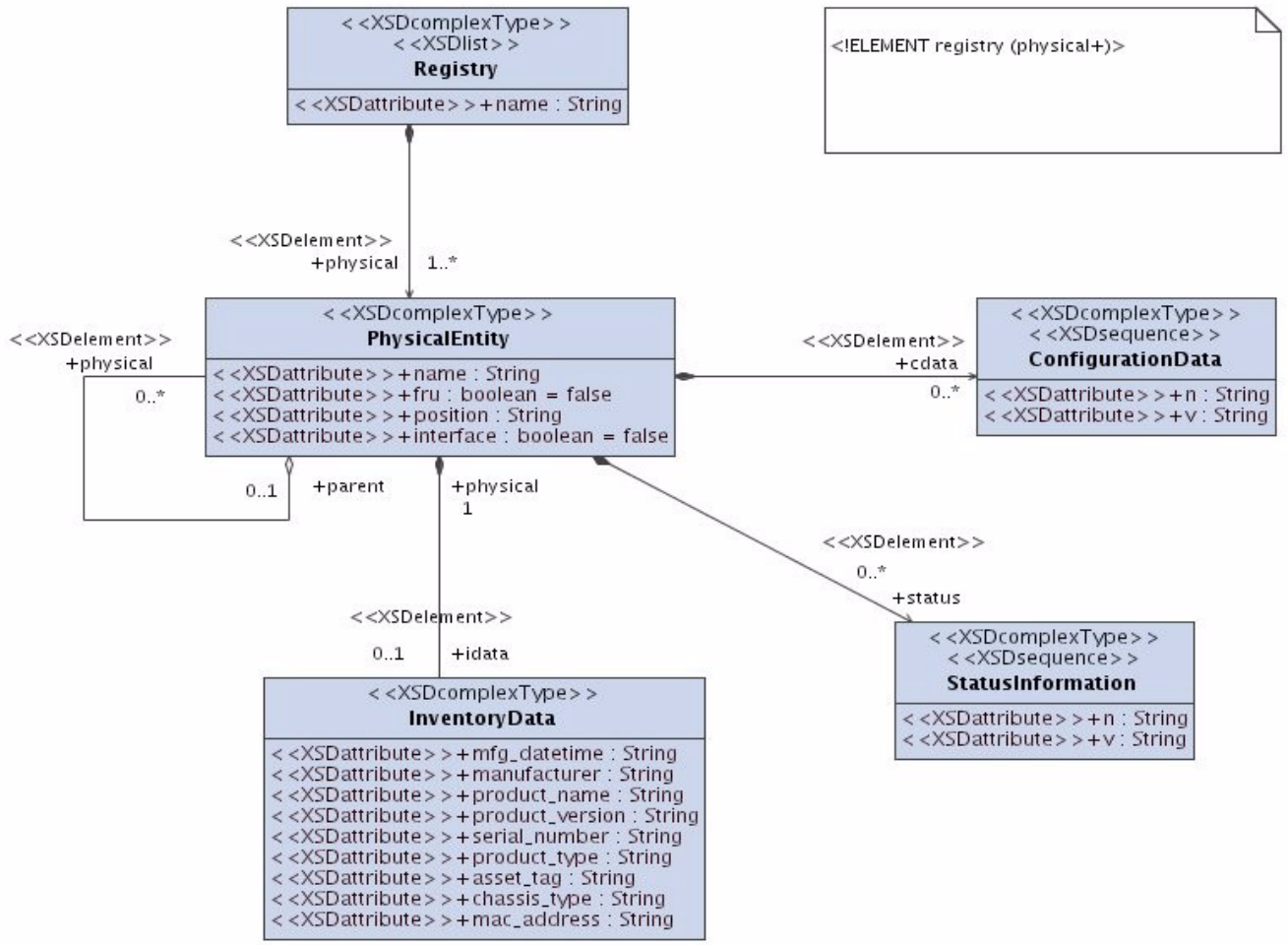


Figure 12.6 - “Registry” part of the “XML for HPI” PSM

12.2 Specific attributes and parameters information

N/A.

12.3 Specific data types

N/A.

12.4 Specific failure codes

N/A.

12.5 Conformance Criteria

This PSM deems all PIM data as mandatory and does not acknowledge any optional profile.

This criteria does not imply that an implementation which conforms to the “XML for HPI” PSM must also have a PIM which conforms to all the compliance profiles. Yet it implies that an implementation of the “XML for HPI” PSM must not refuse as erroneous a file which contains data which are not recognized in the conformance criteria of its PIM.

This criteria is aimed to allow easier exchanges of files among AMSM implementations.

12.5.1 Mapping

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  hardware description consists of an optional inventory,
  a set of configuration definitions, and an optional registry.

  The inventory describles physical elements which may be part
  of the system, or not (typically, it can include the description
  of spare parts.

  The configuration definitions support the description of allowed
  hardware structures, and the definition of mapping of "logical"
  hardware on physical elements.

  The registry defines the actual configuration (possibly discovered
  at startup).
-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="hardware">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" ref="inventory"/>
        <xs:element maxOccurs="unbounded" ref="configuration"/>
        <xs:element minOccurs="0" ref="registry"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <!-- The inventory is a list of physical elements -->
  <xs:element name="inventory">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" ref="physical"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <!--
    A physical element may itself contain other physical elements.
    It has optional inventory data (idata),
                   configuration data (cdata),
                and status data (sdata)
  -->
  <xs:element name="physical">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" ref="idata"/>
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="cdata"/>
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="sdata"/>
        <xs:element maxOccurs="unbounded" ref="physical"/>
      </xs:sequence>
      <xs:attributeGroup ref="attlist.physical"/>
    </xs:complexType>
  </xs:element>
```

```
<!--
   Actual data (inventory, configuration, status) appears as
  attributes of idata, cdata, sdata
-->
<xs:element name="cdata">
  <xs:complexType>
    <xs:attributeGroup ref="attlist.cdata"/>
  </xs:complexType>
</xs:element>
<xs:element name="idata">
  <xs:complexType>
    <xs:attributeGroup ref="attlist.idata"/>
  </xs:complexType>
</xs:element>
<xs:element name="sdata">
  <xs:complexType>
    <xs:attributeGroup ref="attlist.sdata"/>
  </xs:complexType>
</xs:element>
<!--
  Attributes of physical element
  All attributes are stated to be optional. Actually, for a physical
  element enclosed in an enclosing element, either the "name" or "position"
  attribute must be given, and must provide a unique identifier for the
  enclosed element among the elements enclosed in the same enclosing
  element.
-->
<!--
  name - needs to be unique only in the scope of the enclosing
  physical element, if any
-->
<xs:attributeGroup name="attlist.physical">
  <xs:attribute name="name"/>
  <xs:attribute name="fru" default="false">
    <xs:simpleType>
      <xs:restriction base="xs:token">
        <xs:enumeration value="false"/>
        <xs:enumeration value="true"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="position"/>
  <xs:attribute name="interface"/>
</xs:attributeGroup>
<!--
  fru - a boolean attribute set to true if the corresponding physical
  element (and all enclosed elements) are a Field Replaceable Unit
-->
<!-- position - the relative position of the physical element -->
<!--
  interface - means that the element provides an interface to an
  enclosing element
-->
<!--
  Attributes of configuration and status data.
  These are 'name/value' (n,v) couples
-->
<xs:attributeGroup name="attlist.cdata">
  <xs:attribute name="n" use="required"/>
  <xs:attribute name="v" use="required"/>
```

```
</xs:attributeGroup>
<xs:attributeGroup name="attlist.sdata">
  <xs:attribute name="n" use="required"/>
  <xs:attribute name="v" use="required"/>
</xs:attributeGroup>
<!-- Attributes of inventory data. -->
<!-- mfg_datetime - Manufacturing date and time -->
<xs:attributeGroup name="attlist.idata">
  <xs:attribute name="mfg_datatime"/>
  <xs:attribute name="manufacturer"/>
  <xs:attribute name="product_name"/>
  <xs:attribute name="product_version"/>
  <xs:attribute name="serial_number"/>
  <xs:attribute name="product_type"/>
  <xs:attribute name="asset_tag"/>
  <xs:attribute name="chassis_type"/>
  <xs:attribute name="euid"/>
</xs:attributeGroup>
<!-- manufacturer - Manufacturer identification -->
<!-- product_name - Product name -->
<!-- product_version - Product version -->
<!-- serial_number - Serial number -->
<!-- product_type - Product type -->
<!-- asset_tag - asset tag -->
<!--
   chassis_type - Chassis type provided by the element, i.e.,
  3U or 6U Compact PCI, ATCA, Blade, VME, ...
-->
<!-- euid - Extended Unique Identifier, e.g., 48 bits MAC address -->
<!--
  In configuration definitions, the values of attributes of idata
  and the values of the 'v' attribute of cdata are regular expressions;
  A configuration is therefore a pattern to match to actual (discovered)
  configuration
-->
<xs:element name="configuration">
  <xs:complexType>
    <xs:sequence>
      <xs:element maxOccurs="unbounded" ref="physical"/>
      <xs:element maxOccurs="unbounded" ref="logical"/>
    </xs:sequence>
    <xs:attributeGroup ref="attlist.configuration"/>
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.configuration">
  <xs:attribute name="name"/>
</xs:attributeGroup>
<xs:element name="logical">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" ref="grp"/>
      <xs:element minOccurs="0" ref="part"/>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="cdata"/>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="logical"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="registry">
  <xs:complexType>
    <xs:sequence>
```

```
        <xs:element maxOccurs="unbounded" ref="physical"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="grp">
    <xs:complexType/>
  </xs:element>
  <xs:element name="part">
    <xs:complexType/>
  </xs:element>
</xs:schema>
```