Date: November 2009

ALert MAnagement Service (ALMAS) Specification

Version 1.0

OMG Document Number: formal/2009-11-01
Standard document URL: http://www.omg.org/spec/ALMAS/1.0
Associated File(s)*: http://www.omg.org/spec/ALMAS/20090201
http://www.omg.org/spec/ALMAS/20090201/ALMAS.xmi
http://www.omg.org/spec/ALMAS/20090301
http://www.omg.org/spec/ALMAS/20090301/ALMAS_Client.idl
http://www.omg.org/spec/ALMAS/20090301/ALMAS_DataModel.idl
http://www.omg.org/spec/ALMAS/20090301/ALMAS_Management.idl
http://www.omg.org/spec/ALMAS/20090301/ALMAS_Categorisation.xsd
http://www.omg.org/spec/ALMAS/20090301/ALMAS_Configuration.xsd
http://www.omg.org/spec/ALMAS/20090301/ALMAS_Hierarchy.xsd
http://www.omg.org/spec/ALMAS/20090301ALMAS_Template_Alert_Data.xsd
http://www.omg.org/spec/ALMAS/20090301/DDS_Comm_IDLAPM
http://www.omg.org/spec/ALMAS/20090301/DDS_CDPS_IDLAPM
http://www.omg.org/spec/ALMAS/20090301/DDS_DLRL_IDLAPM

---

* original file(s): dtc/09-02-18 (XMI), dtc/09-03-09 (IDL)

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™ , IMM™ , 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 domain of naval Combat Management Systems is characterized by a huge variety of underlying computing platforms, with different and often incompatible means of managing and reporting alerts. Standards-based alert management services are essential for interoperable and open systems. This specification is a standard for ALert MAnagement Service (ALMAS) in CMS systems, consisting of a standard alerts data model and a model for an alert delivery and lifecycle management service.

2 Conformance Criteria

This specification provides a single level of conformance that defines the minimum functionality required of any ALMAS implementation. This is defined as follows:

- PSM level conformance. An ALMAS achieving conformance level 1 conforms to one or more of the middleware platform specific models presented in Chapters 9, 10, and 11 of this document in addition to conforming to the XML Alert template data model and the XML initialization PSMs as presented in sections 8.1 to 8.3 of this document.

In addition the specification identifies a number of classes (postfixed with 'Extensions') together with the categorization PIM and PSM in sections 7.4 and 8.4 which define additional, optional functionality. These can be provided in any combination of the following four options:

1	Support for additional alert cancellation options	ALMASManagerExtensions RemoveAlertsWithDynamicMessageData
2	Support for categorization rules	ALMASManagerExtensions AttachCategorizationRule + DetachCategorizationRule +ALMAS Categorization Rule XML schema
3	Support for more than one language	ALMASResponderExtensions SetLanguage
4	Complete AlertReport text tag substitution	ALMASResponderExtensions GetFilledMessageText

3 Normative References

OASIS Common Alerting Protocol, v1.0,
www.oasis-open.org/committees/download.php/6334/oasis-200402-cap-core-1.0.pdf

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.
Letter of Intent (LOI)	A letter submitted to the OMG BoD's Business Committee signed by an officer of an organization signifying its intent to respond to the RFP and confirming the organization's willingness to comply with OMG's terms and conditions, and commercial availability requirements.
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 Information (RFI)	A general request to industry, academia, and any other interested parties to submit information about a particular technology area to one of the OMG's Technology Committee subgroups.
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 Document

This specification defines the following set of standard terminology:

- An ***event*** is an occurrence that has been detected by the system whose happening must be reported to other members of the system, including human operators.
- An ***alert*** is an entity of observation regarding an event (or sequence of related events) to be reported (directly or indirectly) to an appropriate set of actors.
- ***Alert clients*** are the entities within the system that raise, modify, receive, process, or handle alerts generated by ALMAS.
- An ***alert template*** is a generic definition of a type of alert that can be raised, e.g., 'collision warning' – it requires instantiation to create an alert.
- An ***instance*** of an alert is a specifically raised alert e.g., 'collision warning with track number 111, bearing 020, range 2nm.'

In addition to the general terms defined above, the ALMAS specification includes three main alert categories, as follows:

- Alerts that require no actor action or acknowledgement. This collection of alert templates are generally ***informative*** or routine alerts, they are usually of lower priority / urgency and require some action by ALMAS to be removed.
- Alerts that require acknowledgement by actor(s). This collection of ***acknowledgement*** alert templates is usually more urgent alerts where at least one actor must indicate acknowledgement to ALMAS that the alert has been received.
- Alerts that require both acknowledgement and action confirmation by actor(s). This collection of action alert templates is frequently used for important or critical events where not only is acknowledgement of the receipt required, but also confirmation that the required action has been taken[1].

5 Acronyms and Abbreviations

CMS (Naval) Combat Management System

CORBA Common Object Request Broker Architecture

DCOM Distributed Component Object Model

HTTP HyperText Transfer Protocol

OMG Object Management Group

RFP Request For Proposal

UML Unified Modeling Language

XML eXtensible Mark-up Language

6 Additional Information

6.1 Changes or extensions required to adopted OMG specifications

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

6.2 Acknowledgements

- BAE Systems
- Raytheon Company
- THALES

1. Definition of the required action is not within the scope of ALMAS.

7 Platform Independent Model (PIM)

The PIM has been split into four packages as follows:

- ALMAS Client Callbacks: The interface to be implemented by system components that wish to be notified of ALMAS events such as alerts created, deleted, etc.
- ALMAS Data Model: The structures and their relationships used in an ALMAS system.
- ALMAS Management: Components of the ALMAS system responsible for alert lifecycle oversight.
- ALMAS Configuration: The interfaces and structures for setting up ALMAS.

These are described below, note that ALMAS Categorization is an optional PIM.

7.1 ALMAS Client Callbacks

ALMAS Client Callbacks are the interfaces to be implemented by system components that wish to be notified of ALMAS events such as alerts created, deleted, etc. There are two classes in this package. In order to be plugged into the ALMAS system, a client must implement one of these interfaces, and register with the Alert Manager.

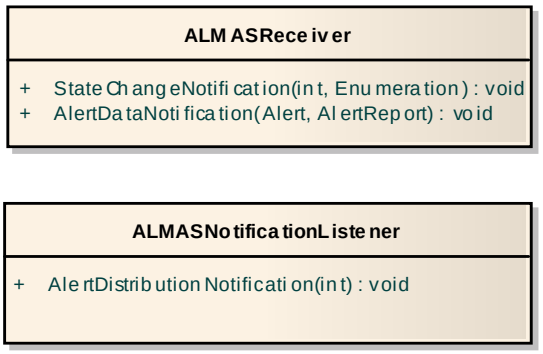

Figure 7.1 - PIM class diagram for ALMAS Clients

7.1.1 ALMASNotificationListener

Class provided by registering notification listeners for receipt of alert distribution notifications.

7.1.1.1 Operation

Name	Type	Summary
AlertDistributionNotification(int)	public void[Parameters]AlertID: int	This is called as soon as a safety critical alert has been received by the ALMAS system. The onward distribution is notified through the StateChangeNotification callback

7.1.2 ALMASReceiver

Class provided by registering alert receivers for provision of the notification callbacks. Only clients that implement this interface and register as receivers can access active alert data. Clients can only register if they are built against the ALMAS interface; therefore no runtime security control is required in this context. Note: The ALMASResponder interface is used to notify ALMAS of “progress” in satisfying the received alert.

7.1.2.1 Operation

Name	Type	Summary
StateChangeNotification(int, Enumeration)	public void[Parameters]AlertID: int,NewState: Enumeration	Indicates a change of state of an alert to a receiver who has registered for this alert's state change notifications. These states are the same states as used in CurrentState for an Alert.
AlertDataNotification(Alert, AlertReport)	public void[Parameters]AlertInfo: Alert,Report: AlertReport	Provides notifications of new and modified alert data.

7.2 ALMAS Data Model

The classes described in this section provide the definition of the contents of Alerts, Alert Templates, and Receivers for ALMAS. The two primary concepts in this data model are of an Alert Template and an Alert. The Alert Template describes the static description of a pre-defined class of alerts, while an Alert contains the specific attributes of a “live” Alert within the ALMAS system. Both utilize the AlertData class to describe many of their field attributes and values.

Note that the constraint called ‘alert_data’ in the figure below is defined as follows:

"context a: Alert inv: if ((a.alert_data.Category = Information) or (a.alert_data.Category = Warning))) then (a.CurrentState <> Handled)"

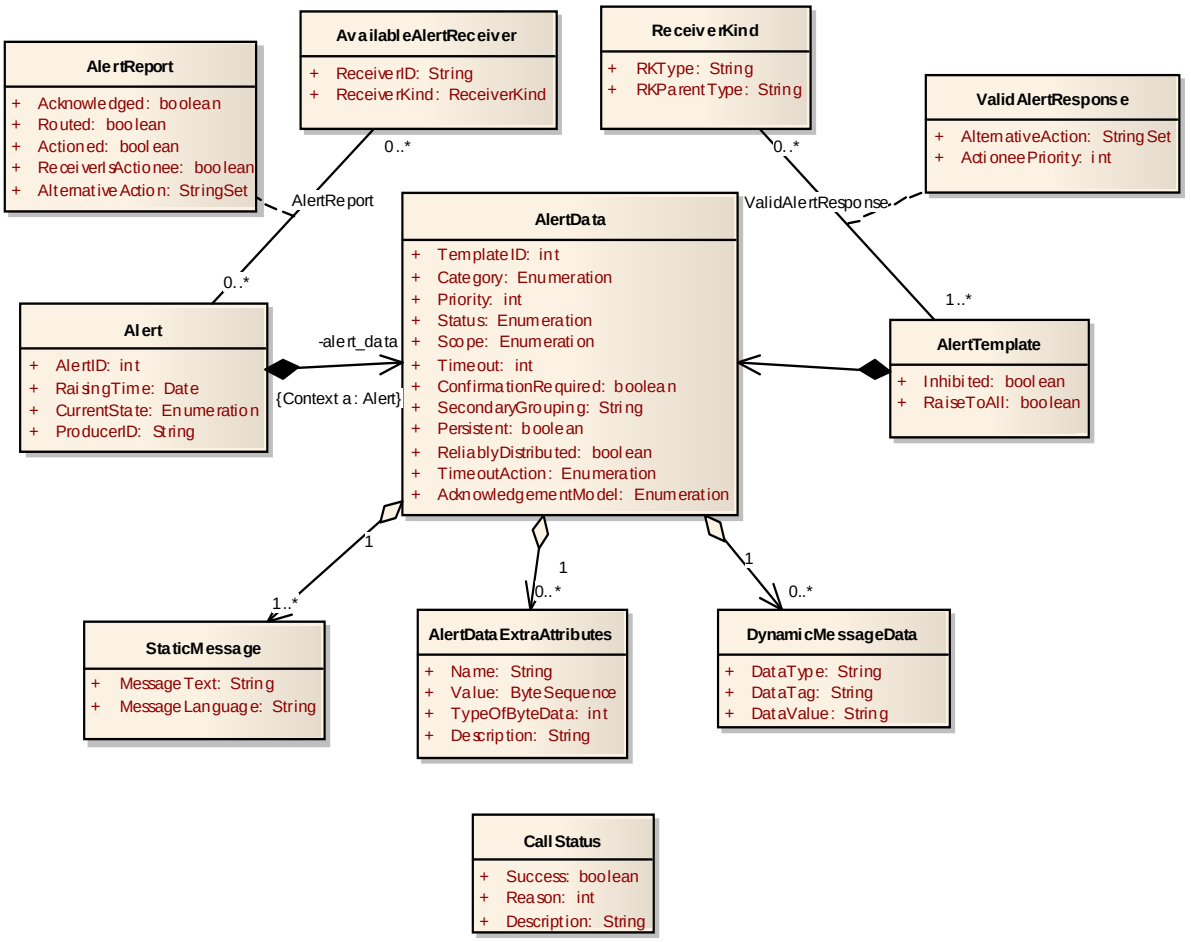

Figure 7.2 - PIM class diagram for ALMAS Data Model

7.2.1 Alert

An active alert within ALMAS. The Alert class provides the main entity that ALMAS uses for tracking the state of an alert. The specific data such as message and other attributes for an active alert is provided in the AlertData class that is a member attribute of the Alert.

7.2.1.1 Attribute

Name	Type	Summary
AlertID	public int	The instance id for the specific instance of the alert.
RaisingTime	public Date	The time at which the alert was raised.
CurrentState	public Enumeration	Holds the current state of the alert, valid states are determined by the category of the alert {Raised, Routed, Received, Acknowledged, Handled, Cancelled, Timed_Out}. Note that Handled is not a valid state for Information and Warning Alerts.
ProducerID	public String	The producer freetext ID - corresponds to CAP source

7.2.2 AlertData

This represents the set of data shared between the alert template and alert classes. All fields have default values that can be changed when alerts are raised/updated. This may be set up through the use of templates as specified through the XML PSM, which initializes AlertTemplate and its associated classes.

7.2.2.1 Attribute

Name	Type	Summary
TemplateID	public int	A unique identifier for template that owns this alert data (or that was used to create the alert if this is referenced from Alert). Valid range from 1 upwards.
Category	public Enumeration	This enumeration can take the value Action / Warning / Information / Situation.
Priority	public int	Alert priority as an integer value in the range 1-99. The priority is open for client use and not intended for interpretation by ALMAS.
Status	public Enumeration	Corresponds to the OASIS CAP Status field. • Actual - Actionable by all targeted recipients. • Exercise- Actionable only by designated exercise participants; exercise identifier should appear in an Alert Data Extra Attributes element. • System - For messages that support alert network internal functions. • Test - Technical testing only, all recipients disregard.
Scope	public Enumeration	Corresponds to CAP scope.

Attribute (continued)

Name	Type	Summary
Timeout	public int	Specifies the time, in seconds, required to elapse before the alert will timeout and perform its default timeout action. 0 implies there is no timeout.
ConfirmationRequired	public boolean	This is set if confirmation of receipt is required, e.g., a safety critical alert requires confirmation that it has been distributed. If this is set to true, the producer has registered for receipt of the distribution notification.
SecondaryGrouping	public String	This is an additional field to support client specific filtering mechanisms.
Persistent	public boolean	Indicates whether the alert data is required to be persistent in the event of a system restart.
ReliablyDistributed	public boolean	A flag which, when true, indicates that the alert should have guaranteed delivery.
TimeoutAction	public Enumeration	This is an enumeration that has values of {cancel, notify, cancel & notify}.
AcknowledgementModel	public Enumeration	Sets the conditions upon which the alert state can transition to 'acknowledged.' This has the options of {none, anyone, all}.

7.2.3 AlertDataExtraAttributes

This is a class representing items of alert data that are specific to particular clients, that require supporting in order to fulfill possible requirements of an alert management system (such as images, screen locations, or other binary data), but are not general enough to be defined explicitly as data types in an ALMAS. Effectively ALMAS provides blind delivery of the information provided by this class to the alert receiver without any knowledge as to its intended meaning and behavior. The extra attributes are configured via the ALMAS Alert definition xml PSM specified in Section 8.1, "The Template Alert Data Specification File," on page 27. If defined in the Alert definition XML provided to ALMAS, then ALMAS shall support the definition, receipt, storage, and passing of this data to receivers as part of a standard implementation.

7.2.3.1 Attribute

Name	Type	Summary
Name	public String	Name of the client specific attribute
Value	public ByteSequence	Contents as a byte sequence
TypeOfByteData	public int	Valid values for this are: 0 = string 1 = Integer8 2 = Integer16 3 = Integer32 4 = Float32 5 = Float64 6 = bytes
Description	public String	This field is used to provide an indication of the content e.g., image (jpg), URL, track object ID, etc.

7.2.4 AlertReport

This provides the delivery message of an Alert to an ALMASReceiver. It contains the Alert and the current status information. This will contain details of whether the instance has been acknowledged by this receiver etc. and will be completed with respect to any dynamic message data.

7.2.4.1 Attribute

Name	Type	Summary
Acknowledged	public boolean	Identified whether the alert has been acknowledged by this receiver.
Routed	public boolean	Identified whether the alert can be confirmed to have been routed as per the 'routed' alert substate.
Actioned	public boolean	Identified whether the alert has been actioned by this receiver.
ReceiverIsActionee	public boolean	Indicates that this receiver is the chosen actionee for this alert.
AlternativeAction	public StringSet	Provides means by which an alternative action outside of the scope of ALMAS can be distributed with the alert via ALMAS.

7.2.5 AlertTemplate

An AlertTemplate specifies the generic characteristics of a specific alert type "at rest" (e.g., the general characteristics of a collision warning alert). This includes the category of alert, such as Action, etc. An AlertTemplate uses an associated AlertData object to specify the contents of the template. An AlertTemplate can be used to specify the properties commonly used within a system. At the time of raising an Alert from a template, the user/system provides the relevant instance data of that alert.

7.2.5.1 Attribute

Name	Type	Summary
Inhibited	public boolean	The inhibition status of that alert type. If this is ‘true,’ then attempts to raise an alert of that type will fail.
RaiseToAll	public boolean	Indicates that the alert should be raised to all available receivers rather than specified ones.

7.2.6 AvailableAlertReceiver

The class used to identify a receiver of alerts. A registered receiver of alerts. The AvailableAlertReceiver is registered with ALMAS through the ALMASResponder API. The AvailableAlertReceiver is directly associated with an ALMASReceiver through the ReceiverID attribute, which is provided at registration time to ALMAS using the RegisterReceiver method.

7.2.6.1 Attribute

Name	Type	Summary
ReceiverID	public String	Unique identifier for the receiver.
ReceiverKind	public ReceiverKind	The kind of the receiver as an explicit attribute link to the Receiver Kind class.

7.2.7 CallStatus

This is the ALMAS a general purpose success/failure descriptor class used throughout ALMAS. If Success, then the other parameters are not applicable.

7.2.7.1 Attribute

Name	Type	Summary
Success	public boolean	Flag indicating pass/fail status
Reason	public int	Enumerated reason correlating to the “Call Status” 0 = Success1 = Not Accepted2 = Malformed Alert3 = Timeout/delivery4 = Requested Service Unavailable5+ = Other
Description	public String	Additional String data further describing status

7.2.8 DynamicMessageData

Since Alerts often have variable data fields, the DynamicMessageData class provides the means for inserting variable content into the Alert’s MessageText during runtime. Replacement values for the DataTag are treated as strict string substitution within the MessageText of the StaticMessage associated with the Alert. This is used to capture the triplet of data tag type, tag position in the alert message and the value that this tag in the template message text should be replaced with.

Note – If the text specified in the StaticMessage contains multiple replacement points (specified by %%t1 through %%tn), then an equal number of DynamicMessageData objects are required for full substitution.

7.2.8.1 Attribute

Name	Type	Summary
DataType	public String	The type of related object e.g., freetext, track, vehicle, position, etc.
DataTag	public String	This identifies the insertion point for the related object in the MessageText associated with the Alert. If the MessageText is "xxxxx %t1 yyyyyyy zzzz," then DataTag has the value 't1.' It is a case sensitive, alphanumeric string.
DataValue	public String	The value of the object instantiation. Given a type of string to be general enough to support free text and track/vehicle id's alike.

7.2.9 ReceiverKind

The descriptor of an alert receiver. This could, for example, be an operator role. ReceiverKind (RK) objects are used in many places in ALMAS including the specification of what operators/clients will receive which Alerts.

- These are used to show all possible receivers of an Alert, when used in an AlertTemplate.
- These are used during runtime to identify the actual receivers for an active alert.

7.2.9.1 Attribute

Name	Type	Summary
RKType	public String	String identifier of the kind of receiver, for example the role of a receiving operator.
RKParentType	public String	The hierarchical parent receiver kind name that this one "belongs to." This is used by ALMAS to resolve cases where a specific RK is not available but handing is required by an appropriate receiver. Note that a lack of a Parent is indicated by an empty string.

7.2.10 StaticMessage

Provides the default message text for an alert as a tuplet of the actual static text and the language in which the text is provided. If the StaticMessage requires runtime updating, then use data tags as specified in DynamicMessageData.

7.2.10.1 Attribute

Name	Type	Summary
MessageText	Public String	This is a text string, which in an Alert or AlertTemplate is only partially completed. With the MessageText being “xxxxx %t1 yyyyyyy zzzz” in an Alert or AlertTemplate, and with a DynamicMessageData with DataTag having the value ‘t1’ and DataValue having the value ‘123,’ then the resulting MessageText in response to GetFilledMessageText will be ‘xxxxx 123 yyyyyyy zzzz.’ All substitution points are bracketed by use of “<space>%” and <space>, and are case sensitive, alphanumeric strings (“t1” in the above) which should correspond to a DataTag in an associated DynamicMessageData.
MessageLanguage	public String	The message ‘Locale’

7.2.11 ValidAlertResponse

The ValidAlertResponse is the association class that specifies the list of actions that a particular ReceiverKind (e.g., “role”) can take in response to an Alert of an AlertTemplate type. It also specifies the “pecking order” of that ReceiverKind among all ReceiverKinds associated with that AlertTemplate.

The set of alternative action strings can be used by the system to provide a constrained set of “command-response” options to the client. For example, ValidAlertResponses for an “Engagement Request Alert” might include “WILCO,” “CANTCO,” etc.

7.2.11.1 Attribute

Name	Type	Summary
AlternativeAction	public StringSet	The 'names' of alternative actions available to the relevant actor.
ActioneePriority	public int	The priority of the ReceiverKind as actionee for a specific alert as described by its template. The highest priority actionee for an action alert should be chosen as the current actionee for the alert. This will then flow into the ReceiverIsActionee field of the AlertReport.

7.3 ALMAS Management

This section describes the classes responsible for raising, routing, maintaining the state of, and destroying alerts through their lifecycle. ALMAS uses a collection of specialized component interfaces for maintaining state, data, and lifecycle of Alerts. In general, systems that utilize ALMAS will interact during runtime primarily through the ALMAS Producer, Responder, and Notification Listener classes. The ALMAS Manager interface is utilized more at system startup.

Deleting alert instances is under the control of ALMAS itself as part of its lifecycle management, and not at the request of its users. In more detail:

- Any alert is removed when cancelled. Note that Situation alerts are only removed when cancelled.
- Information and Warning alerts are removed when the required number of acknowledgements (as identified in the AlertData AcknowledgementModel attribute) are given or (if a timeout is defined) when the timeout is expired.

- Action alerts are removed when HandleAlert is called by the Receiver identified as the Actionee in its AlertReport.

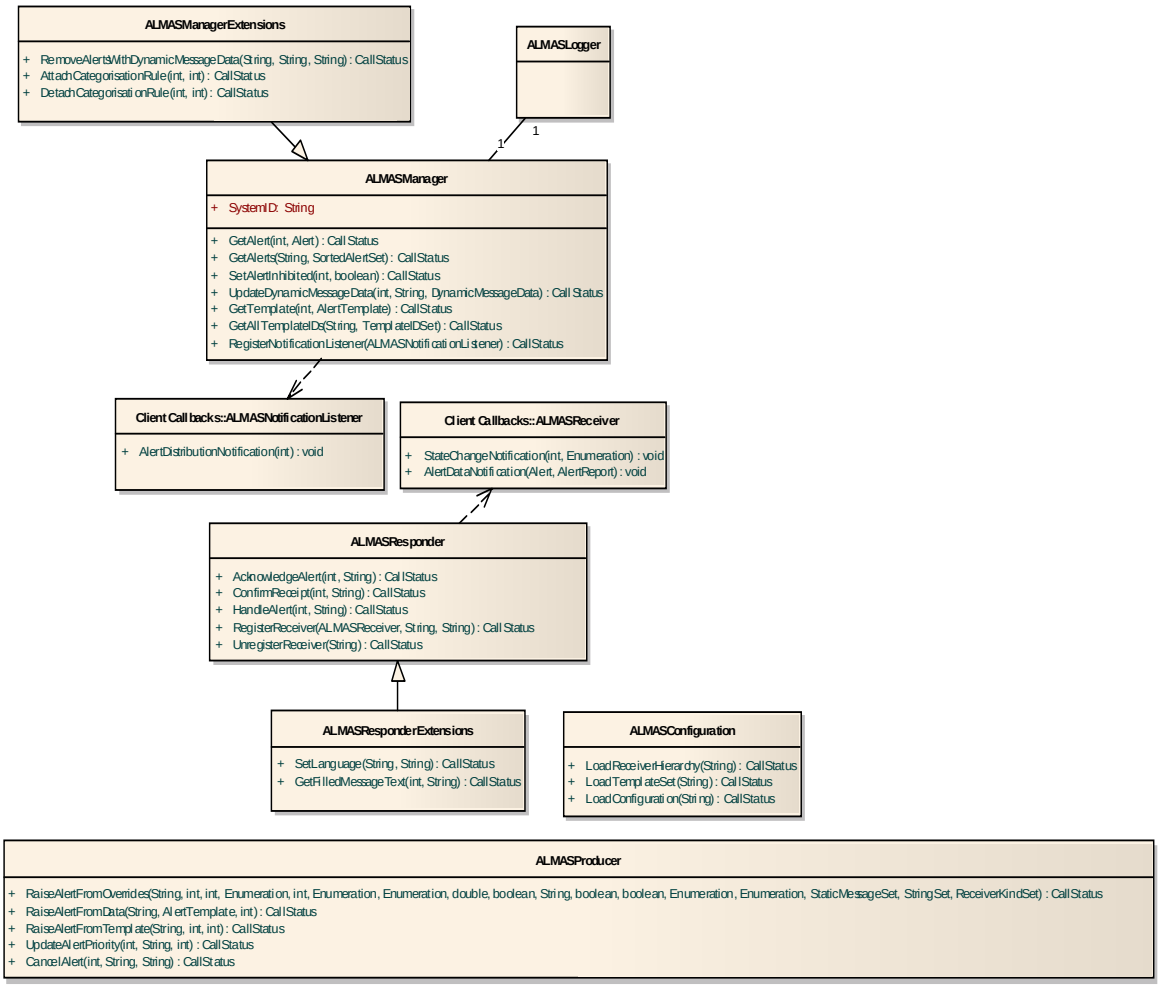

Figure 7.3 - PIM class diagram for ALMAS Management

This package provides the main API to the ALMAS service.

7.3.1 ALMASConfiguration

Provides an API by which systems can configure ALMAS to behave in a more tailored manner in order to satisfy very specific requirements. There are three categories of configuration file that can be used by ALMAS: the receiver hierarchy, templates, and configuration information. The string filename is expected to resolve to either a local file accessible to ALMAS, or a URL accessible to ALMAS. The returned CallStatus object from each of the methods provides an indication of success/failure and any additional relevant rationale describing that status.

7.3.1.1 Operation

Name	Type	Summary
LoadReceiverHierarchy(String)	public CallStatus[Parameters]Filename: String	Loads the receiver hierarchy as provided by the client via xml conforming to the relevant xml schema document. The specification of the ReceiverHierarchy file format can be found in Section 8.3.
LoadTemplateSet(String)	public CallStatus[Parameters]Filename: String	Loads a template set into the ALMAS database. Multiple calls to this method result in the union of the new templates with the existing templates in ALMAS. The specification of the template file format can be found in Section 8.1.
LoadConfiguration(String)	public CallStatus[Parameters]Filename: String	Loads the ALMAS configuration file as provided by the client. The specification of the configuration file format can be found in Section 8.2.

7.3.2 ALMASLogger

Logging mechanism to record historical Alert information. This version of the ALMAS Standard does not specify a specific interface to/from the ALMAS logger; however, conformant ALMAS implementations must include logging of alerts raised, delivered, received, handled, and cleared.

7.3.3 ALMASManager

The ALMASManager interface provides the minimal set of APIs necessary to track ALMAS activity. Additionally, the ALMASManager provides the interface in ALMAS for retrieving Alerts and AlertTemplates, and registering for the notification of delivery of Alerts. Note that the registration of receivers is done via the ALMAS Responder class.

Note – The methods found in the ALMASProducer interface allow the system to update the status or attributes of an alert during runtime.

7.3.3.1 Attribute

Name	Type	Summary
SystemID	private String	Provides a field for specifying the current instance of ALMAS. Corresponds to CAP sender.

7.3.3.2 Operation

Name	Type	Summary
GetAlert(int, Alert)	public CallStatus[Parameters]AlertID: int,out Alert: Alert	Retrieves data for a specific raised alert from ALMAS given the passed AlertID. Assumes the requestor knows the AlertID to retrieve.
GetAlerts(String, SortedAlertSet)	public CallStatus[Parameters]Filter: String,out AlertSet: SortedAlertSet	Retrieves a set of all alert instances within ALMAS that satisfy the filter. The filter string provided will be compared with the value in the AlertData SecondaryGrouping field. All matches will be returned in the Set.
SetAlertInhibited(int, boolean)	public CallStatus[Parameters] TemplateID: int,Inhibition: boolean	Sets the inhibition status of a specific alert template to suppress or allow the raising of all alerts of that template.
UpdateDynamicMessageData(int, String, DynamicMessageData)	public CallStatus[Parameters]AlertID: int,ObjectValue: String,OldData: Dynamic MessageData	Indicates a change to the value of a related object for the provided alert ID. OldData is necessary in order to clearly indicate which dynamic message data should be changed.
GetTemplate(int, AlertTemplate)	public CallStatus[Parameters] TemplateID: int,out Template: AlertTemplate	Retrieves an existing alert template from ALMAS by providing the template ID.
GetAllTemplateIDs(String, TemplateIDSet)	public CallStatus[Parameters]Filter: String,out TemplateIDs: TemplateIDSet	Retrieves all Alert Template IDs, or if the Filter string is non-null, it returns those which satisfy the Filter. The filter string provided will be compared with the value in the AlertData SecondaryGrouping field. All matches will be returned in the Set.
RegisterNotification Listener(ALMASNotification Listener)	public CallStatus[Parameters]Handle: ALMASNotificationListener	Registers a new Notification Listener for receipt of the alert distribution notifications.

7.3.4 ALMASManagerExtensions

This class contains optional extensions to the alert manager functionality. These extensions may or may not be implemented in simple ALMAS implementations.

7.3.4.1 Operation

Name	Type	Summary
RemoveAlertsWithDynamicMessage Data(String, String)	public CallStatus[Parameters]CancellerID: String,DataType: String,DataValue: String	Indicates to ALMAS that a specific real world object has been removed, and therefore all associated alerts are no longer valid. These will then be deleted from ALMAS.Implementation is optional.
AttachCategorisationRule(int, int)	public CallStatus[Parameters]TemplateID: int,RuleID: int	Associates a categorization rule with an AlertTemplate
DetachCategorisationRule(int, int)	public CallStatus[Parameters]TemplateID: int,RuleID: int	Disassociates a categorization rule from an AlertTemplate

7.3.5 ALMAS Producer

Provides the API by which system objects producing alerts can create and update alerts that are generated. A CallStatus object will be returned to indicate whether the request has been accepted by ALMAS. If a system wished to track the lifecycle of the alert, they must implement the NotificationListener functionality to receive updates.

Three mechanisms by which alerts can be raised are provided by the ALMASProducer interface class. Two variants RaiseAlertFromTemplate and RaiseAlertFromOverrides allow the system to raise an alert by simply specifying the alert ID, template ID and their own ProducerID, one of these also allows the over-ride of any placeholders that may be present in the ‘Message’ attribute of the alert data class associated with that template. The raiser may also optionally override any of the following parameters: Message, MessageLanguage, Category, Status, Scope, Timeout, ConfirmationRequired, AlertReceiverSet, Priority, TimeoutAction, and AcknowledgementModel.

The RaiseAlertFromData method allows the raiser to specify a completely new alert with no basis on any existing templates. Systems using ALMAS may not wish to support alert templates depending on their size, complexity and level of alert usage, in which case that system can always use RaiseAlertFromData without need to instantiate any templates at any point during operation.

The status or attributes of an alert can be updated during runtime by calling the UpdateAlert method found in the ALMASProducer interface. The ALMASProducer then works with the ALMAS system to ensure state and data is properly maintained in the system.

7.3.5.1 Operation

Name	Type	Summary
RaiseAlertFromOverrides(String, int, int, Enumeration, int, Enumeration, Enumeration, double, boolean, String, boolean, boolean, Enumeration, Enumeration, StaticMessageSet, StringSet, ReceiverKindTypeSet)	public CallStatus[Parameters]ProducerID: String,TemplateID: int,out AlertID: int,Category: Enumeration,Priority: int,Status: Enumeration,Scope: Enumeration,Timeout: double,ConfirmationRequired: boolean,SecondaryGrouping: String, Persistent: boolean, ReliablyDistributed: boolean, TimeoutAction: Enumeration,AcknowledgementModel: Enumeration, StaticMessages: StaticMessageSet,DynamicMessageDataSet: StringSet,AlertReceivers: ReceiverKindTypeSet	This will cause an alert based on a known alert template to be created and raised.ProducerID, TemplateID, and the out parameter AlertID are mandatory, all other parameters are optional. Return parameter indicates success or failure reason.
RaiseAlertFromData(String, AlertTemplate, int)	public CallStatus[Parameters]ProducerID: String,AlertInfo: AlertTemplate,out AlertID: int	Raise an alert not present in the ALMAS template database. A temporary AlertTemplate is created (whose TemplateID is ignored), to facilitate the creation.Return parameter indicates success or failure reason.
RaiseAlertFromTemplate(String, int, int)	public CallStatus[Parameters]ProducerID: String,TemplateID: int,out AlertID: int	Raise an alert without any of the optional parameters for optimal use in the normal case.
UpdateAlertPriority(int, String, int)	public CallStatus[Parameters]AlertID: int,ProducerID: String,Priority: int	Update an existing raised alert instances priority.
CancelAlert(int, String, String)	public CallStatus[Parameters]AlertID: int,CancellerID: String,CancellationReason: String	Cancel a specific alert within ALMAS Return parameter indicates success or failure reason.

7.3.6 ALMASResponder

Provides the API for systems to respond to and provide feedback to ALMAS about alerts received. Embedded in this class are the methods to register and un-register your system-specific receiver.

The system notifies ALMAS through this interface of significant events that have occurred to change the state of an alert.

7.3.6.1 Operation

Name	Type	Summary
AcknowledgeAlert(int, String)	public CallStatus[Parameters]AlertID: int,ReceiverID: String	Indication from an alert receiver that they have acknowledged receipt of the alert and no longer require distribution of its information.
ConfirmReceipt(int, String)	public CallStatus[Parameters]AlertID: int,ReceiverID: String	Confirmation by an alert receiver that they have successfully received the alert to ensure reliable distribution. The ReceiverID field enables action & situation alerts to transition when sufficient confirmations have been received. 'Sufficient' is the 'actionee' for action alerts, and anyone for situation alerts. It can also be used for logging purposes.
HandleAlert(int, String)	public CallStatus[Parameters]AlertID: int,ReceiverID: String	Indication from an Alert Receiver that they have performed the appropriate action required by an Action alert and that the alert can therefore be removed from ALMAS as no longer applicable.
RegisterReceiver(ALMAS Receiver, String, String)	public CallStatus[Parameters]ReceiverHandler: ALMASReceiver,ReceiverID: String,RKType: String	This registers a receiver with ALMAS, the parameters are ReceiverHandle (for callback); ReceiverID (for use in all other methods, including UnregisterReceiver) and RKType to provide link to RK hierarchy.
UnregisterReceiver(String)	public CallStatus[Parameters]Receiv-erID: String	Removes a registered receiver from ALMAS, indicating that they are no longer available for receipt of alert data.

7.3.7 ALMASResponderExtensions

Optional extensions to the alert responder functionality.

7.3.7.1 Operation

Name	Type	Summary
SetLanguage(String, String)	public CallStatus[Parameters]ReceiverID: String,Language: String	Sets the language that this specific receiver should see their message text displayed in where appropriate.
GetFilledMessageText(int, String)	public CallStatus[Parameters]AlertID: int,out MessageText: String	Returns the message text post related info substitutions. This is an optional helper function as the client could derive this itself.

7.4 Alert Categorization

The Alert Categorization PIM allows the expression of Event-Condition-Action rules that can guide automatic triggering of alerts. This represents an optional part of the specification, as it is also possible to trigger alerts through the ALMAS API. The Categorization PIM allows for the implementation of monitoring components (agents) that can trigger alerts based on different events taking place in the system, such as time events or changes in the internal state of the system.

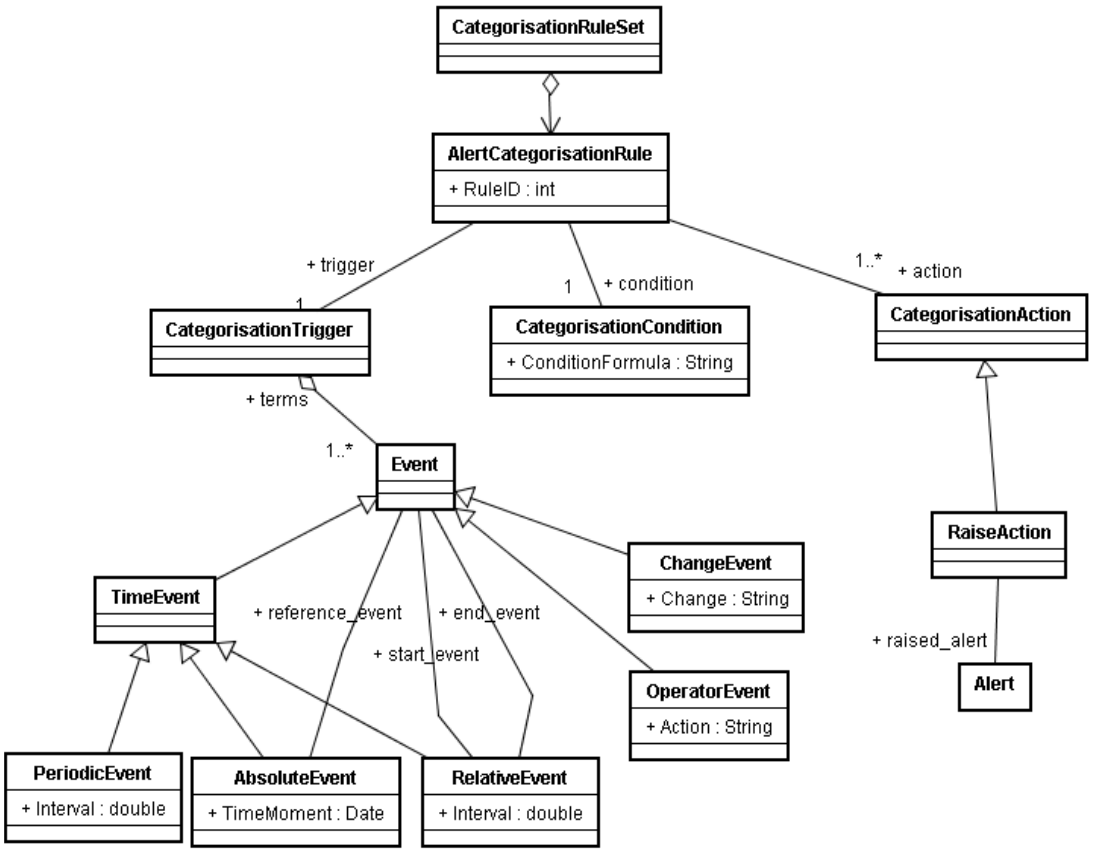

Figure 7.4 - Alert Categorization Platform Independent Model

7.4.1 AbsoluteEvent

Represents an event taking place once at a specific time moment.

7.4.1.1 Attribute

Name	Type	Summary
TimeMoment	public Date	The time of the trigger event

7.4.2 AlertCategorizationRule

Alert Categorization Rule represents an Event-Condition-Action rule guiding the categorization. On Event being triggered, a Condition is evaluated. If it evaluates to true, the corresponding Categorization Action is executed.

7.4.2.1 Attribute

Name	Type	Summary
RuleID	public int	The rule identifier

7.4.3 CategorizationAction

Categorization Action represents the action to be executed when an event has occurred and the conditions required have been fulfilled.

7.4.4 CategorizationCondition

The Categorization Condition represents the condition part of the Event, Condition Action rule.

7.4.4.1 Attribute

Name	Type	Summary
ConditionFormula	public String	The condition formula

7.4.5 CategorizationRuleSet

This is the set of Event, Condition Action rules that apply to this ALMAS system.

7.4.6 CategorizationTrigger

The Categorization Trigger represents the Event, which is able to be observed by ALMAS that can trigger categorization.

7.4.7 ChangeEvent

One type of event such as enter/leave area, change of generic data value, etc.

7.4.7.1 Attribute

Name	Type	Summary
Change	public String	The change which is required

7.4.8 Event

General class of Event, used within the Categorization Trigger.

7.4.9 OperatorEvent

Operator initiated events, for example operator changing a role.

7.4.9.1 Attribute

Name	Type	Summary
Action	public String	The operator action required

7.4.10 PeriodicEvent

Represents a relative event, i.e., an event taking place at a specific (time) interval after another event.

7.4.10.1 Attribute

Name	Type	Summary
Interval	public double	The condition formula

7.4.11 RaiseAction

A kind of Categorization Action that raises an alert. Other categorization actions could be added.

7.4.12 RelativeEvent

Represents a periodic event taking place between start_event and end_event at a specific periodicity (interval).

7.4.12.1 Attribute

Name	Type	Summary
Interval	public double	Time interval after the reference_interval event at which the RelativeEvent is to take place.

7.4.13 Time Event

A timeout event, which can be absolute, relative, or periodic.

7.5 Dynamic Behavior

7.5.1 Action Situation Alert State Model

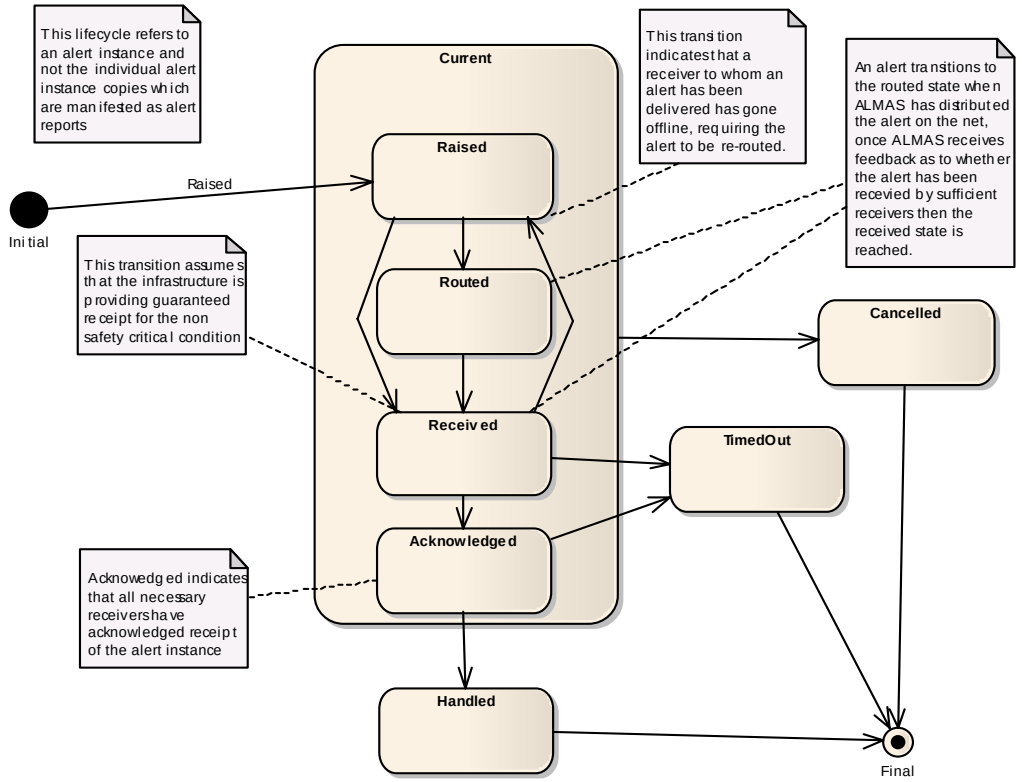

Figure 7.5 - Action/Situation Alert Lifecycle

7.5.2 Information Warning Alert State Model

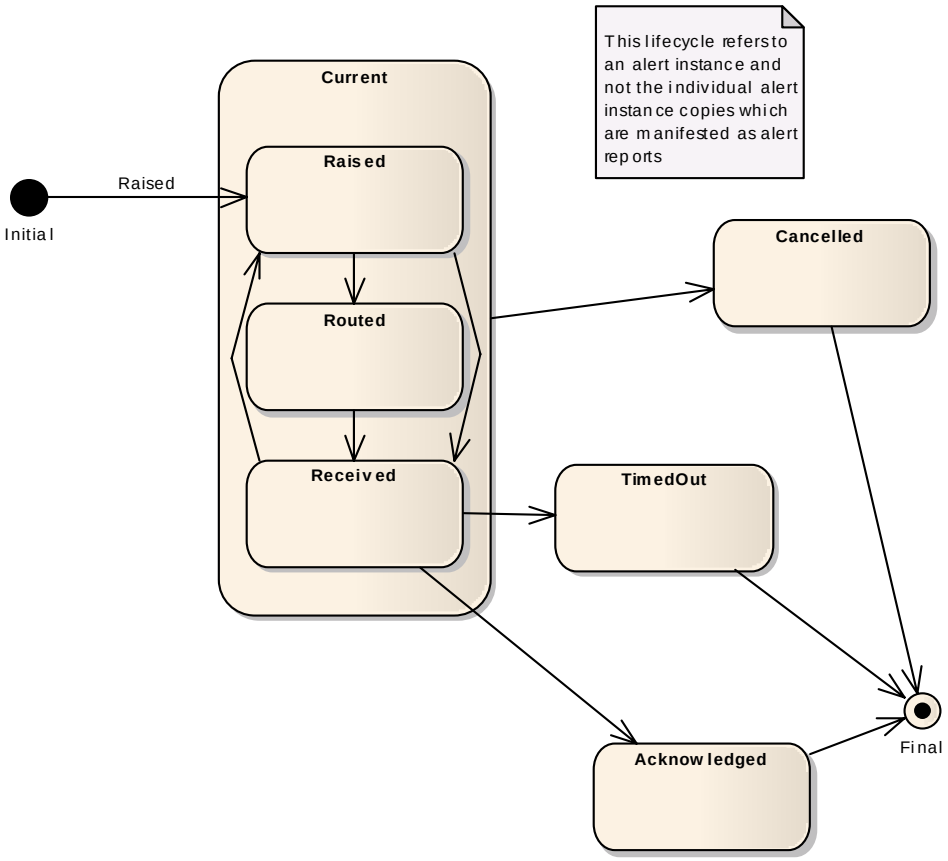

Figure 7.6 - Information/Warning Alert Instance Lifecycle

7.5.3 Alert Registration and Creation

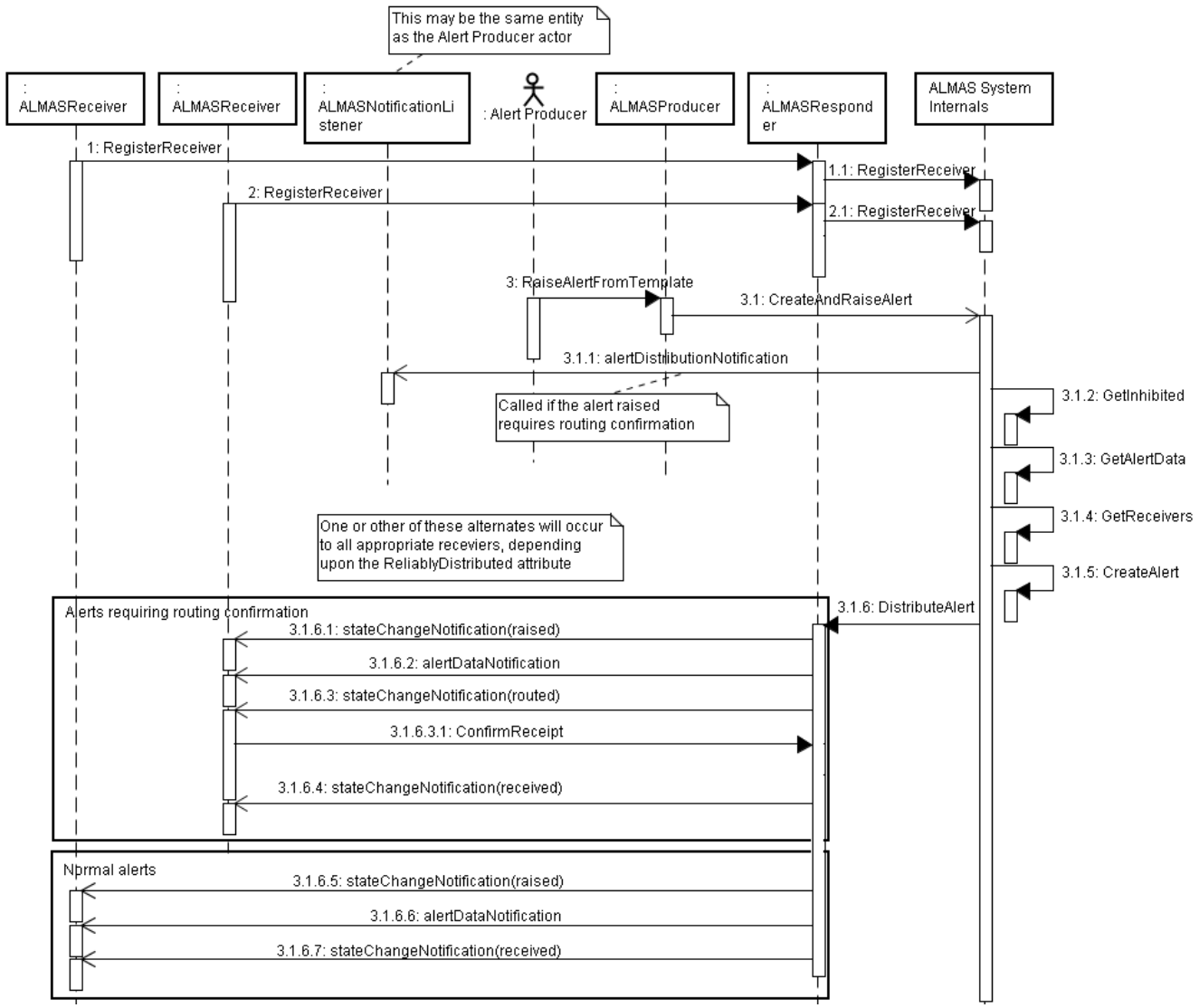

Figure 7.7 - Alert Registration and Creation Sequence Diagram

The above sequence diagram shows the interaction with the ALMAS service from several user perspectives.

First it indicates the receiver registration interactions (shown as threads 1 and 2 in the figure).

Second it shows the alert raising interactions from an alert producer, with an illustration of the additional callback made if the alert requires routing confirmation (thread 3 up to 3.1.1).

Interactions 3.1.2 through 3.1.6 are indications of the internal activities, but are not requirements upon the internals (hence shown under the fictional class ALMAS System Internals).

Finally interactions 3.1.6.1-4 and 3.1.6.5-7 are two possible interaction from ALMAS back to the alert receiver, depending upon the ReliablyDistributed attribute of the alert. In the case of this attribute being TRUE then 3.1.6.1-4 are executed, otherwise 3.1.6.5-7 are executed.

8 XML Platform Specific Model (PSM)

8.1 The Template Alert Data Specification File

The Template Alert Data specification file is an xml schema document that specifies the ontology of the alert template data to be loaded into an ALMAS by the LoadTemplateSet method. Use of this is therefore effectively optional but any client that wishes to make use of templates may do so by supplying corresponding valid xml for loading into the system.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Alert Data Template schema -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attribute-
FormDefault="unqualified" version="1.0a" id="Alert_Template_Data">
  <xs:element name="Alert_Template_Root" type="Alerts_Templates_T">
    <xs:annotation>
      <xs:documentation>Root element containing Alert Template Data.</xs:documentation>
    </xs:annotation>
    <xs:unique name="Template_Id">
      <xs:selector xpath="./Alert_Template"/>
      <xs:field xpath="Template_Id"/>
    </xs:unique>
  </xs:element>
  <xs:complexType name="Alerts_Templates_T">
    <xs:sequence>
      <xs:element name="Alert_Template" type="Alerts_Template_T" minOccurs="0" maxOc-
curs="unbounded">
        <xs:annotation>
          <xs:documentation>The template of an alert.</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="Alerts_Template_T">
    <xs:sequence>
      <xs:element name="Template_Id">
        <xs:simpleType>
          <xs:annotation>
            <xs:documentation>The unique template identifier.</xs:documentation>
          </xs:annotation>
          <xs:restriction base="xs:integer">
            <xs:minInclusive value="1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="Alert_Category">
```

```
    <xs:simpleType>
      <xs:annotation>
        <xs:documentation>Enumeration of Alert Category.</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:enumeration value="Action"/>
        <xs:enumeration value="Situation"/>
        <xs:enumeration value="Information"/>
        <xs:enumeration value="Warning"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
  <xs:element name="Alert_Default_Priority">
    <xs:simpleType>
      <xs:restriction base="xs:integer">
        <xs:minInclusive value="1"/>
        <xs:maxInclusive value="99"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
  <xs:element name="Status">
    <xs:simpleType>
      <xs:annotation>
        <xs:documentation>OASIS CAP Derived Status</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:enumeration value="Actual"/>
        <xs:enumeration value="Exercise"/>
        <xs:enumeration value="System"/>
        <xs:enumeration value="Test"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
  <xs:element name="Scope">
    <xs:simpleType>
      <xs:annotation>
        <xs:documentation>OASIS CAP Derived Scope</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:enumeration value="PublicScope"/>
        <xs:enumeration value="RestrictedScope"/>
        <xs:enumeration value="PrivateScope"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
  <xs:element name="Timeout">
```

```
        <xs:simpleType>
          <xs:annotation>
            <xs:documentation>Time until alert timeout in seconds, where 0 indicates no timeout
required</xs:documentation>
          </xs:annotation>
          <xs:restriction base="xs:integer">
            <xs:minInclusive value="0"/>
            <xs:maxInclusive value="3600"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="ConfirmationRequired" type="xs:boolean"/>
      <xs:element name="Secondary_Grouping" minOccurs="0">
        <xs:simpleType>
          <xs:annotation>
            <xs:documentation>Secondary grouping for filtering aid</xs:documentation>
          </xs:annotation>
          <xs:restriction base="xs:string"/>
        </xs:simpleType>
      </xs:element>
      <xs:element name="Persistent" type="xs:boolean"/>
      <xs:element name="ReliablyDistributed" type="xs:boolean"/>
      <xs:element name="TimeoutAction">
        <xs:simpleType>
          <xs:annotation>
            <xs:documentation>The action to be performed upon alert timeout</xs:documentation>
          </xs:annotation>
          <xs:restriction base="xs:string">
            <xs:enumeration value="CancelOnly"/>
            <xs:enumeration value="NotifyOnly"/>
            <xs:enumeration value="CancelWithNotify"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="AcknowledgementModel">
        <xs:simpleType>
          <xs:annotation>
            <xs:documentation>Required acknowledgement profile before progressing the alert to
'Acknowledged'</xs:documentation>
          </xs:annotation>
          <xs:restriction base="xs:string">
            <xs:enumeration value="AckByNone"/>
            <xs:enumeration value="AckByAnyone"/>
            <xs:enumeration value="AckByAll"/>
          </xs:restriction>
        </xs:simpleType>
```

```
      </xs:element>
      <xs:element name="Inhibited" type="xs:boolean" minOccurs="0"/>
      <xs:element name="Raise_To_All" type="xs:boolean"/>
      <xs:element name="Static_Message" type="Static_Message_T" maxOccurs="unbounded"/>
      <xs:element name="Alert_Data_Extra_Attributes" type="Alert_Data_Extra_Attributes_T" minOc-
curs="0"
maxOccurs="unbounded"/>
      <xs:element name="Dynamic_Message_Data" type="Dynamic_Message_Data_T" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="Alert_Routing" type="Alert_Routing_T" minOccurs="0" maxOc-
curs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="Static_Message_T">
    <xs:sequence>
      <xs:element name="MessageText">
        <xs:simpleType>
          <xs:annotation>
            <xs:documentation>The Alert Template Text</xs:documentation>
          </xs:annotation>
          <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="MessageLanguage">
        <xs:simpleType>
          <xs:annotation>
            <xs:documentation>The alert locale</xs:documentation>
          </xs:annotation>
          <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="Alert_Data_Extra_Attributes_T">
    <xs:sequence>
      <xs:element name="Name">
        <xs:simpleType>
          <xs:annotation>
            <xs:documentation>The Attribute Name</xs:documentation>
          </xs:annotation>
          <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
```

```
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="TypeOfByteData">
        <xs:simpleType>
          <xs:annotation>
            <xs:documentation>Flag to indicate the type of data</xs:documentation>
          </xs:annotation>
          <xs:restriction base="xs:integer">
            <xs:minInclusive value="0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="Description">
        <xs:simpleType>                                <xs:annotation>
            <xs:documentation>Description of contents e.g. image(jpg), URL, Track report etc</xs:docu-
mentation>                                             </xs:annotation>
          <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
          </xs:restriction>
        </xs:simpleType>
      <xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="Dynamic_Message_Data_T">
    <xs:sequence>
      <xs:element name="Variable_Type">
        <xs:simpleType>
          <xs:annotation>
            <xs:documentation>Type of variable data</xs:documentation>
          </xs:annotation>
          <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="Tag">
        <xs:annotation>
          <xs:documentation>The position of the data item within message</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:minInclusive value="1"/>
            <xs:maxInclusive value="20"/>
        </xs:restriction>
        </xs:simpleType>
```

```
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="Alert_Routing_T">
    <xs:sequence>
      <xs:element name="Receiver_Kind">
        <xs:annotation>
          <xs:documentation>A receiver kind</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="AlternativeAction" minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation>A none standard alert response</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="Actionee_Priority">
        <xs:annotation>
          <xs:documentation>The priority of the actionee to deal with this alert</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:integer">
            <xs:minInclusive value="1"/>
            <xs:maxInclusive value="10"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

8.2 The ALMAS Configuration File

The ALMAS configuration file is an xml schema document specifying some client specific attributes to allow an ALMAS to be more flexible to a clients specific needs from their ALMAS implementation. This should allow for greater interoperability and usability. It is loaded by use of the LoadConfiguration method.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- ALMAS Configuration -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.0a" id="ALMAS_Configuration_Data">
  <xs:element name="ALMAS_Config_Root" type="Alerts_Config_T">
    <xs:annotation>
      <xs:documentation>Root element containing ALMAS Configuration Data.</xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:complexType name="Alerts_Config_T">
    <xs:sequence>
      <xs:element name="Max_No_Alerts">
        <xs:annotation>
          <xs:documentation>Maximum number of alerts in the system</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:integer">
            <xs:minInclusive value="0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="Max_No_Alerts_For_Receiver">
        <xs:annotation>
          <xs:documentation>Maximum number of alerts for each receiver</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:integer">
            <xs:minInclusive value="0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

8.3 The Receiver Hierarchy Configuration File

The Receiver Hierarchy configuration file specifies the structure of the relationships between alert receivers to allow for resilience processing in the event of receiver non-availability. If an alert requires routing to a specific receiver who is not available, then the receiver Hierarchy file specifies a parent receiver that can be used in place of the one originally specified.

Iterative progression up the hierarchy can then be possible until an available receiver is found in place of the original one.

The receiver hierarchy is loaded via the LoadReceiverHierarchy method.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Receiver Hierarchy schema -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.0a" id="Receiver_Hierarchy_Data">
  <xs:element name="Receiver_Hierarchy_Root" type="Receiver_Hierarchy_T">
    <s:annotation>
      <xs:documentation>Root element containing Hierarchy Data.</xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:complexType name="Receiver_Hierarchy_T">
    <xs:sequence>
      <xs:element name="Receiver_Kind" type="Receiver_Kind_T" maxOccurs="unbounded">
      <xs:annotation>
          <xs:documentation>A Receiver Kind</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="Receiver_Kind_T">
    <xs:sequence>
      <xs:element name="Type">
        <xs:annotation>
          <xs:documentation>The receiver kind e.g. SPS</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:minLength value="1" />
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="ParentType">
        <xs:annotation>
          <xs:documentation>The 'type' of the receiver kind's parent e.g. TPS</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:minLength value="1" />
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

8.4 The ALMAS Categorization Rule File

The Categorization Rule file is an xml schema document that specifies the categorization rules, which can be attached to (or detached from) alerts by means of AttachCategorizationRule method in ALMAS Manager. The configuration file is read by an ALMAS implementation at startup, but attaching/detaching of rules to alerts can be done dynamically at runtime using those methods.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Categorisation_Rule_Set" type="Categorisation_Rule_Set"/>
  <xs:complexType name="Categorisation_Rule_Set">
    <xs:sequence>
      <xs:element name="Alert_Categorisation_Rule" type="Alert_Categorisation_Rule"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="Alert_Categorisation_Rule" type="Alert_Categorisation_Rule"/>
  <xs:complexType name="Alert_Categorisation_Rule">
    <xs:sequence>
      <xs:element name="ruleID" type="xs:int"/>
      <xs:element name="action" type="Categorisation_Action" maxOccurs="unbounded"/>
      <xs:element name="condition" type="Categorisation_Condition"/>
      <xs:element name="trigger" type="Categorisation_Trigger"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="Categorisation_Trigger" type="Categorisation_Trigger"/>
  <xs:complexType name="Categorisation_Trigger">
    <xs:sequence>
      <xs:element name="terms" type="Event" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="RelativeEvent" type="RelativeEvent"/>
  <xs:complexType name="RelativeEvent">
    <xs:complexContent>
      <xs:extension base="TimeEvent">
        <xs:sequence>
          <xs:element name="interval" type="xs:double"/>
          <xs:element name="reference_event" type="Event"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="PeriodicEvent" type="PeriodicEvent"/>
  <xs:complexType name="PeriodicEvent">
    <xs:complexContent>
      <xs:extension base="TimeEvent">
        <xs:sequence>
```

```
          <xs:element name="interval" type="xs:double"/>
          <xs:element name="start_event" type="Event"/>
          <xs:element name="end_event" type="Event"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="AbsoluteEvent" type="AbsoluteEvent"/>
  <xs:complexType name="AbsoluteEvent">
    <xs:complexContent>
      <xs:extension base="TimeEvent">
        <xs:sequence>
          <xs:element name="time_moment" type="xs:date"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="Categorisation_Action" type="Categorisation_Action"/>
  <xs:complexType name="Categorisation_Action">
    <xs:sequence/>
  </xs:complexType>
  <xs:element name="Categorisation_Condition" type="Categorisation_Condition"/>
  <xs:complexType name="Categorisation_Condition">
    <xs:sequence>
      <xs:element name="condition_formula" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="Raise_Action" type="Raise_Action"/>
  <xs:complexType name="Raise_Action">
    <xs:complexContent>
      <xs:extension base="Categorisation_Action">
        <xs:sequence>
          <xs:element name="raised_alert" type="Alert"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="Alert" type="Alert"/>
  <xs:complexType name="Alert">
    <xs:sequence/>
  </xs:complexType>
  <xs:element name="Event" type="Event"/>
  <xs:complexType name="Event">
    <xs:sequence/>
  </xs:complexType>
  <xs:element name="TimeEvent" type="TimeEvent"/>
```

```
  <xs:complexType name="TimeEvent">
    <xs:complexContent>
      <xs:extension base="Event">
        <xs:sequence/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="Operator_Event" type="Operator_Event"/>
  <xs:complexType name="Operator_Event">
    <xs:complexContent>
      <xs:extension base="Event">
        <xs:sequence>
          <xs:element name="action" type="xs:string"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="Change_Event" type="Change_Event"/>
  <xs:complexType name="Change_Event">
    <xs:complexContent>
      <xs:extension base="Event">
        <xs:sequence>
          <xs:element name="change" type="xs:string"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:schema>
```

9 OMG CORBA IDL Platform Specific Model

9.1 Rationale

The objective of this PSM is to normalize the CORBA/IDL structures and interfaces. This PSM aims to support the entire PIM interface.

In order for this interface to be reasonably compatible with the DDS PSM, also provided, the data model part is separated from the functional interface model.

All attributes, methods, and associations are mapped to IDL elements. As a general rule, therefore, classes with methods are mapped to CORBA/IDL interfaces, classes without methods are mapped to structs, attributes are mapped to CORBA/IDL attributes, associations to read only attributes, and methods to methods that deal with errors through CORBA exceptions.

Subscribe methods and indication classes are also mapped within a client IDL file that has to be implemented by clients in order to receive indications (i.e., callbacks) from ALMAS.

9.2 ALMAS Data Model IDL

```
// Copyright 2005-2008 THALES, BAE Systems, Raytheon

#include "timebase.idl"
#ifndef __ALMAS_DataModel_DEF
#define __ALMAS_DataModel_DEF
#pragma prefix "omg.org"

module ALMAS_DataModel {

 typedef long ALMAS_AlertIDType;

 typedef long ALMAS_TemplateIDType;

 typedef long ALMAS_TimeoutType;

 typedef TimeBase::TimeT ALMAS_DateTimeType; // EVoT compatible  long long

 typedef sequence<octet> ALMAS_ByteSequence;

 typedef sequence<string> ALMAS_StringSet;

 enum ALMAS_CategoryType {
   Action,
   Warning,
   Information,
   Situation};
```

```
enum ALMAS_StateType {
  Raised,
  Routed,
  Received,
  Acknowledged,
  Handled,
  Cancelled,
  TimedOut};

eum ALMAS_StatusType {
  Actual,
  Exercise,
  System,
  Test};

enum ALMAS_ScopeType {
  PublicScope,
  RestrictedScope,
  PrivateScope};

enum ALMAS_TimeoutActionType {
  CancelOnly,
  NotifyOnly,
  CancelWithNotify};

enum ALMAS_AckModelType {
  AckByNone,
  AckByAnyone,
  AckByAll};

struct ALMAS_CallStatus {
  boolean Success;
  short Reason;
  string Description; };

struct ALMAS_ValidAlertResponseType {
  ALMAS_StringSet AlternativeAction;
  short ActioneePriority; };

struct ALMAS_ReceiverKindType {
  string RKType;
  string RKParentType;
  ALMAS_ValidAlertResponseType ValidResponse; };
typedef sequence<ALMAS_ReceiverKindType> ALMAS_ReceiverKindTypeSet;
```

```
struct ALMAS_DynamicMessageDataType {
  string DataType;
  string DataTag;
  string DataValue; };
typedef sequence<ALMAS_DynamicMessageDataType> ALMAS_DynamicMessageDataTypeSet;

struct ALMAS_StaticMessageType {
  string MessageText;
  string MessageLanguage; };
typedef sequence<ALMAS_StaticMessageType> ALMAS_StaticMessageTypeSet;

struct ALMAS_AlertDataExtraAttributesType {
  string Name;
  short TypeOfByteData;
  string Description;
   ALMAS_ByteSequence Value; };
typedef sequence<ALMAS_AlertDataExtraAttributesType> ALMAS_AlertDataExtraAttributesTypeSet;

struct ALMAS_AlertDataType {
  ALMAS_TemplateIDType TemplateID;
  ALMAS_CategoryType Category;
  short Priority;
  ALMAS_StatusType Status;
  ALMAS_ScopeType Scope;
  ALMAS_TimeoutType Timeout;
  boolean ConfirmationRequired;
  string SecondaryGrouping;
  boolean Persistent;
  boolean ReliablyDistributed;
  ALMAS_TimeoutActionType TimeoutAction;
  ALMAS_AckModelType AcknowledgementModel;
  ALMAS_StaticMessageTypeSet StaticMessages;
  ALMAS_DynamicMessageDataTypeSet DynamicMessages;
  ALMAS_AlertDataExtraAttributesTypeSet ExtraAttributes; };

struct ALMAS_AlertTemplateType {
  boolean Inhibited;
  boolean RaiseToAll;
  ALMAS_AlertDataType AlertData;
  ALMAS_ReceiverKindTypeSet ReceiverKinds; };

struct ALMAS_AlertReportType {
  boolean Acknowledged;
  boolean Routed;
  boolean Actioned;
  boolean ReceiverIsActionee;
```

```
  ALMAS_StringSet AlternativeAction;
  string ReceiverID;
  ALMAS_AlertIDType AlertID; };

 struct ALMAS_AvailableAlertReceiverType {
  string ReceiverID;
  ALMAS_ReceiverKindType ReceiverKind; };
 typedef sequence<ALMAS_AvailableAlertReceiverType> ALMAS_AvailableAlertReceiverTypeSet;

 struct ALMAS_Alert {
  ALMAS_AlertIDType AlertID;
  ALMAS_DateTimeType RaisingTime;
  ALMAS_StateType CurrentState;
  string ProducerID;
  ALMAS_AlertDataType AlertData;
  ALMAS_AvailableAlertReceiverTypeSet Receivers; };
};

#endif
```

9.3 ALMAS Client IDL

```
// Copyright 2005-2008 THALES, BAE Systems, Raytheon

#include "ALMAS_DataModel.idl"
#ifndef __ALMAS_Client_DEF
#define __ALMAS_Client_DEF
#pragma prefix "omg.org"

module ALMAS_Client {

 interface ALMAS_Receiver {

  oneway void StateChangeNotification (
   in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
   in ALMAS_DataModel::ALMAS_StateType NewState);

  oneway void AlertDataNotification ( // alert ID is embedded within info
   in ALMAS_DataModel::ALMAS_Alert AlertInfo,
   in ALMAS_DataModel::ALMAS_AlertReportType Report);
 };

 interface ALMAS_NotificationListener {

  oneway void AlertDistributionNotification (
   in ALMAS_DataModel::ALMAS_AlertIDType AlertID);
```

```
  };
};

#endif
```

9.4 ALMAS Management IDL

```
// Copyright 2005-2008 THALES, BAE Systems, Raytheon

#include "ALMAS_Client.idl"
#include "ALMAS_DataModel.idl"
#ifndef __ALMAS_Management_DEF
#define __ALMAS_Management_DEF
#pragma prefix "omg.org"

module ALMAS_Management {

 typedef sequence<ALMAS_DataModel::ALMAS_Alert> ALMAS_AlertSet;

 typedef sequence<ALMAS_DataModel::ALMAS_TemplateIDType> ALMAS_TemplateIDTypeSet;

 interface ALMAS_Manager {

  attribute string ALMAS_SystemID;

  // alert retrieval methods

  ALMAS_DataModel::ALMAS_CallStatus GetAlert (
    in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
    out ALMAS_DataModel::ALMAS_Alert Alert);

  ALMAS_DataModel::ALMAS_CallStatus GetAlerts (
    in string Filter,
    out ALMAS_AlertSet AlertSet);

  // ALMAS-wide control methods

  ALMAS_DataModel::ALMAS_CallStatus SetAlertInhibited (
    in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
    in boolean Inhibition);

  ALMAS_DataModel::ALMAS_CallStatus UpdateDynamicMessageData (
    in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
    in string ObjectValue,
    in ALMAS_DataModel::ALMAS_DynamicMessageDataType OldValue);
```

```
 ALMAS_DataModel::ALMAS_CallStatus RegisterNotificationListener (
   in ALMAS_Client::ALMAS_NotificationListener Handle);

 // Template management methods

 ALMAS_DataModel::ALMAS_CallStatus GetTemplate (
   in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
   out ALMAS_DataModel::ALMAS_AlertTemplateType Template);

 ALMAS_DataModel::ALMAS_CallStatus GetAllTemplateIDs (
   in string Filter,
   out ALMAS_TemplateIDTypeSet TemplateIDSet);
};

interface ALMAS_ManagerExtensions : ALMAS_Manager {

 ALMAS_DataModel::ALMAS_CallStatus RemoveAlertsWithDynamicData (
   in string CancellerID,
   in string DataType,
   in string DataValue);

 AMAS_DataModel::ALMAS_CallStatus AttachCategorisationRule (
   in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
   in long RuleID);

 ALMAS_DataModel::ALMAS_CallStatus DetachCategorisationRule (
   in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
   in long RuleID);
 };
interface ALMAS_Producer {

 ALMAS_DataModel::ALMAS_CallStatus RaiseAlertFromOverrides (
   in string ProducerID,
   in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
   in ALMAS_DataModel::ALMAS_CategoryType Category,
   in boolean ValidCategory,
   in short Priority,
   in boolean ValidPriority,
   in ALMAS_DataModel::ALMAS_StatusType Status,
   in boolean ValidStatus,
   in ALMAS_DataModel::ALMAS_ScopeType Scope,
   in boolean ValidScope,
   in ALMAS_DataModel::ALMAS_TimeoutType Timeout,
   in boolean ValidTimeout,
   in boolean ConfirmationRequired,
   in boolean ValidConfirmationRequired,
```

```
    in string SecondaryGrouping,
    in boolean ValidSecondaryGrouping,
    in boolean Persistent,
    in boolean ValidPersistent,
    in boolean ReliablyDistributed,
    in boolean ValidReliablyDistributed,
    in ALMAS_DataModel::ALMAS_TimeoutActionType TimeoutAction,
    in boolean ValidTimeoutAction,
    in ALMAS_DataModel::ALMAS_AckModelType AcknowledgementModel,
    in boolean ValidAcknowledgementModel,
    in ALMAS_DataModel::ALMAS_StaticMessageSet StaticMessages,
    in boolean ValidStaticMessages,
    in ALMAS_DataModel::ALMAS_DynamicMessageDataTypeSet DynamicMessageData,
    in boolean ValidDynamicMessageData,
    in ALMAS_DataModel::ALMAS_ReceiverKindTypeSet AlertReceivers,
    in boolean ValidAlertReceiverSet,
    out ALMAS_DataModel::ALMAS_AlertIDType AlertID);

   ALAS_DataModel::ALMAS_CallStatus RaiseAlertFromData (
    in string ProducerID,
    in ALMAS_DataModel::ALMAS_AlertTemplateType AlertInfo,
    out ALMAS_DataModel::ALMAS_AlertIDType AlertID);

  AMAS_DataModel::ALMAS_CallStatus RaiseAlertFromTemplate (
    in string ProducerID,
    in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
    out ALMAS_DataModel::ALMAS_AlertIDType AlertID);

  ALMAS_DataModel::ALMAS_CallStatus UpdateAlertPriority (
    in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
    in string ProducerID,
    in short Priority);

  AMAS_DataModel::ALMAS_CallStatus CancelAlert (
    in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
    in string CancellerID,
    in string CancellationReason);
};

interface ALMAS_Responder {

  ALMAS_DataModel::ALMAS_CallStatus RegisterReceiver (
    in ALMAS_Client::ALMAS_Receiver ReceiverHandle,
    in string ReceiverID,
    in string RKType);
```

```
 ALMAS_DataModel::ALMAS_CallStatus UnregisterReceiver (
   in string ReceiverID);

 AMAS_DataModel::ALMAS_CallStatus AcknowledgeAlert (
   in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
   in string ReceiverID);

 ALMAS_DataModel::ALMAS_CallStatus HandleAlert (
   in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
   in string ReceiverID);

 ALMAS_DataModel::ALMAS_CallStatus ConfirmReceipt (
   in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
   in string ReceiverID);
};

interface ALMAS_ResponderExtensions : ALMAS_Responder {

 ALMAS_DataModel::ALMAS_CallStatus SetLanguage (
   in string ReceiverID,
   in string Language);

 ALMAS_DataModel::ALMAS_CallStatus GetFilledMessageText (
   in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
   out string MessageText);
};

interface ALMAS_Configuration {

 ALMAS_DataModel::ALMAS_CallStatus LoadReceiverHierarchy (
   in string Filename );

 ALMAS_DataModel::ALMAS_CallStatus LoadTemplateSet (
   in string Filename );

 ALMAS_DataModel::ALMAS_CallStatus LoadConfiguration (
   in string Filename );
 };
};

#endif
```

10 DDS/DCPS Platform Specific Model

10.1 Rationale

The approach in this PSM is to compare it to the CORBA PSM and highlight differences as necessary. In the DDS PSM two (not exclusive) ways are provided for modeling the management module:

- DCPS-only mapping, in which interfaces are modeled as topics (singletons) and methods as pairs of (request- and reply) topics.
- DLRL mapping, which models classes and methods more directly. The mapping is based on information provided by PrismTech on DLRL data modeling. This entails following when compared to the CORBA PSM:
  - use of valuetypes instead of interfaces – note that a valuetype that is to be distributed by DLRL must inherit from DDS::ObjectRoot.
  - there must be an XML-based mapping from DLRL to DCPS. This mapping is not provided in the specification as it is expected that the default DLRL-DCPS mapping is used.

A DCPS-only implementation will use only DCPS-only mapping, while a DLRL implementation will use a combination of DCPS and DLRL mappings.

All topics are identified by the #pragma keylist immediately after them. Submitters are aware that this is not a DDS standard construct (this is a product-specific OpenSplice facility) and will revise the specification when there is a standardized way of declaring keys.

10.1.1 DCPS Level Mapping

A generic response topic is used for responses to all method calls; note that this does not provide return values, but just the error code.

Return values are implemented in DCPS by publication of an appropriate topic.

In terms of mapping the PIM-level methods on DCPS, the following rules are applied:

- Wherever possible, PIM-level methods are mapped to subscriptions or publications of respective DDS topics. This means that even though these methods cannot be found in the DDS PSM IDL, they can be executed on the PSM level by simply calling the required function from the DDS API. For example, the method GetAlert in ALMAS Manager can therefore be implemented by a DDS read of the Alert topic, with attached condition to receive only the Alert with the ID we are interested in.
- In all other cases, so-called “control topics” are used (such as also applied in the AMSM specification). The names of the topics identify the method that they realize. The control topics include also an identifier of the request (assumed to be uniquely generated by the calling application). The responses to methods are modeled as instances of topic ALMAS_Response, 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.

Additionally there is a singleton topic for ALMAS_Manager as this has attributes.

It is assumed that request IDs are generated by the caller and that they are unique. In this case the caller is responsible for matching. This is in alignment with the approach taken in AMSM.

10.2 ALMAS Data Model - shared

```
// copyright 2005-8 THALES, BAE Systems, Raytheon

// #include "timebase.idl"
#include "dds_dcps.idl"  // use for DDS standard compatible time types

#ifndef __ALMAS_DataModel_DEF
#define __ALMAS_DataModel_DEF

module ALMAS_DataModel {

 typedef long long ALMAS_AlertIDType;

 typedef long ALMAS_TemplateIDType;

 typedef long ALMAS_TimeoutType;

 // typedef TimeBase::TimeT ALMAS_DateTimeType; // EVoT compatible – long long
 typedef DDS::Time_t ALMAS_DateTimeType; // DDS compatible

 typedef sequence<octet> ALMAS_ByteSequence;

 typedef sequence<string> ALMAS_StringSet;

 enum ALMAS_CategoryType {
   Action,
   Warning,
   Information,
   Situation};

 enum ALMAS_StateType {
   Raised,
   Routed,
   Received,
   Acknowledged,
   Handled,
   Cancelled
   Timed_Out};

 enum ALMAS_StatusType {
   Actual,
   Exercise,
   System,
   Test};
```

```
enum ALMAS_ScopeType {
  PublicScope,
  RestrictedScope,
  PrivateScope};

enum ALMAS_TimeoutActionType {
  CancelOnly,
  NotifyOnly,
  CancelWithNotify};

enum ALMAS_AckModelType {
  AckByNone,
  AckByAnyone,
  AckByAll};

struct ALMAS_CallStatus {
  boolean Success;
  short Reason;
  string Description; };

struct ALMAS_ValidAlertResponseType {
  ALMAS_StringSet AlternativeAction;
  short ActioneePriority; };

struct ALMAS_ReceiverKindType {
  string RKType;
  string RKParentType;
  ALMAS_ValidAlertResponseType ValidResponse; };
typedef sequence<ALMAS_ReceiverKindType> ALMAS_ReceiverKindTypeSet;

struct ALMAS_DynamicMessageDataType {
  string DataType;
  string DataTag;
  string DataValue; };
typedef sequence<ALMAS_DynamicMessageDataType> ALMAS_DynamicMessageDataTypeSet;

struct ALMAS_StaticMessageType {
  string MessageText;
  string MessageLanguage; };
typedef sequence<ALMAS_StaticMessageType> ALMAS_StaticMessageTypeSet;

struct ALMAS_AlertDataAttributesType {
  ALMAS_CategoryType Category;
  short Priority;
  ALMAS_StatusType Status;
  ALMAS_ScopeType Scope;
```

```
  ALMAS_TimeoutType Timeout;
  boolean ConfirmationRequired;
  string SecondaryGrouping;
  boolean Persistent;
  boolean ReliablyDistributed;
  ALMAS_TimeoutActionType TimeoutAction;
  ALMAS_AckModelType AcknowledgementModel;
  ALMAS_StaticMessageTypeSet StaticMessages;
  ALMAS_DynamicMessageDataTypeSet DynamicMessages; };

struct ALMAS_AlertDataExtraAttributesType {
  string Name;
  short TypeOfByteData;
  string Description;
  ALMAS_ByteSequence Value; };
typedef sequence<ALMAS_AlertDataExtraAttributesType> ALMAS_AlertDataExtraAttributesTypeSet;

struct ALMAS_AlertDataType {
  ALMAS_TemplateIDType TemplateID;
  ALMAS_AlertDataAttributesType Attributes;
  ALMAS_AlertDataExtraAttributesTypeSet ExtraAttributes; };

struct ALMAS_AlertTemplateType {
  boolean Inhibited;
  boolean RaiseToAll;
  ALMAS_AlertDataType AlertData;
  ALMAS_ReceiverKindTypeSet ReceiverKinds; };
#pragma keylist ALMAS_AlertTemplateType AlertData.TemplateID

struct ALMAS_AlertReportType {
  boolean Acknowledged;
  boolean Routed;
  boolean Actioned;
  boolean ReceiverIsActionee;
  ALMAS_StringSet AlternativeAction;
  tring ReceiverID;
  ALMAS_AlertIDType AlertID; };
#pragma keylist ALMAS_AlertReportType ReceiverID, AlertID

struct ALMAS_AvailableAlertReceiverType {
  string ReceiverID;
  ALMAS_ReceiverKindType ReceiverKind; };
typedef sequence<ALMAS_AvailableAlertReceiverType> ALMAS_AvailableAlertReceiverTypeSet;

struct ALMAS_Alert {
  ALMAS_AlertIDType AlertID;
```

```
  ALMAS_DateTimeType RaisingTime;
  ALMAS_StateType CurrentState;
  string ProducerID;
  ALMAS_AlertDataType AlertData;
  ALMAS_AvailableAlertReceiverTypeSet Receivers; };
 #pragma keylist ALMAS_Alert AlertID

};

#endif
```

10.3 DCPS

10.3.1 ALMAS Client

The ALMAS client module is not required in the DDS PSM since this is all available through the use of the standard DDS mechanisms and the topics already defined for ALMAS_StateType and ALMAS_Alert.

10.3.2 ALMAS Management

The following table provides explanation of the mapping of methods in the ALMAS Management module. Only those methods that are mapped directly to DDS level constructs are listed in the table, all methods that are mapped on “control topics” are listed in the subsequent IDL file.

Class (PIM level)	Method	DDS mapping
ALMAS Manager	GetAlert(int, Alert)	DDS read with query condition
ALMAS Manager	GetAlerts(String, SortedAlertSet)	DDS read with query condition
ALMAS Manager	GetTemplate(int)	DDS read with query condition.
ALMAS Manager	GetAllTemplateIDs(String, TempalteIDSet)	DDS read with query condition.
ALMAS Manager	RegisterNotificationListener(ALMAS Notification Listener)	Creation of a new DDS Listener.

```
// copyright 2005-8 THALES, BAE Systems, Raytheon

#include "ALMAS_DataModel.idl"

#ifndef __ALMAS_Management_DEF
#define __ALMAS_Management_DEF

module ALMAS_Management {

 typedef sequence<ALMAS_DataModel::ALMAS_Alert> ALMAS_AlertSet;
```

```
struct ALMAS_Response {
  long long request_id;
  ALMAS_DataModel::ALMAS_CallStatus error_code; };
#pragma keylist ALMAS_Response request_id

// Need a singleton topic for ALMAS_Manager since it has attributes

struct ALMAS_Manager {
  string SystemID;};
#pragma keylist ALMAS_Manager

struct ALMAS_RaiseAlertFromTemplate {
  long long request_id;
  string ProducerID;
  ALMAS_DataModel::ALMAS_TemplateIDType TemplateID; };
#pragma keylist ALMAS_RaiseAlertFromTemplate request_id

struct ALMAS_RegisterReceiver {
  long long request_id;
  string ReceiverID;
  string RKType; };
#pragma keylist ALMAS_RegisterReceiver request_id

struct ALMAS_UnregisterReceiver {
  long long request_id;
  string ReceiverID; };
#pragma keylist ALMAS_UnregisterReceiver request_id

struct ALMAS_RaiseAlertFromOverrides  {
  long request_id;
  string ProducerID;
  ALMAS_DataModel::ALMAS_TemplateIDType TemplateID;
  ALMAS_DataModel::ALMAS_AlertDataAttributesType Attributes;
  boolean CategoryValid;
  boolean PriorityValid;
  boolean StatusValid;
  boolean ScopeValid;
  boolean TomeoutValid;
  boolean ConfirmationRequiredValid;
  boolean SecondaryGroupingValid;
  boolean PersistentValid;
  boolean ReliablyDistributedValid;
  boolean TimeoutActionValid;
  boolean AcknowledgementModelValid;
  boolean StaticMessagesValid;
  boolean DynamicMessagesValid; };
```

```
#pragma keylist ALMAS_RaiseAlertFromOverrides request_id

struct ALMAS_RaiseAlertFromData  {
  long long request_id;
  string ProducerID;
  ALMAS_DataModel::ALMAS_AlertTemplateType AlertInfo; };
#pragma keylist ALMAS_RaiseAlertFromData request_id

 struct ALMAS_CreatedAlert {
  long long request_id;
  ALMAS_DataModel::ALMAS_AlertIDType AlertID; };
#pragma keylist ALMAS_CreatedAlert request_id

struct ALMAS_UpdateAlertPriority {
  long long request_id;
  string ProducerID;
  ALMAS_DataModel::ALMAS_AlertIDType AlertID;
  short Priority; };
#pragma keylist ALMAS_UpdateAlertPriority request_id

struct ALMAS_CancelAlert  {
  long long request_id;
  string CancelerID;
  ALMAS_DataModel::ALMAS_AlertIDType AlertID;
  string CancellationReason; };
#pragma keylist ALMAS_CancelAlert request_id

struct ALMAS_AcknowledgeAlert {
  long long request_id;
  ALMAS_DataModel::ALMAS_AlertIDType AlertID;
  string ReceiverID;};
#pragma keylist ALMAS_AcknowledgeAlert request_id

struct ALMAS_HandleAlert {
  long long request_id;
  ALMAS_DataModel::ALMAS_AlertIDType AlertID;
  string ReceiverID;};
#pragma keylist ALMAS_HandleAlert request_id

struct ALMAS_ConfirmReceipt {
  long long request_id;
  ALMAS_DataModel::ALMAS_AlertIDType AlertID;
  string ReceiverID;};
#pragma keylist ALMAS_ConfirmReceipt request_id

struct ALMAS_SetLanguage {
```

```
  long long request_id;
  string ReceiverID;
  string Language;};
 #pragma keylist ALMAS_SetLanguage request_id

 struct ALMAS_GetFilledMessageText {
  long long request_id;
  ALMAS_DataModel::ALMAS_AlertIDType AlertID;
  string ReceiverID;};
#prgma keylist ALMAS_GetFilledMessageText request_id

 struct ALMAS_FilledMessageText {
  long long request_id;
  ALMAS_DataModel::ALMAS_StringSet Messages; };
 #pragma keylist ALMAS_FilledMessageText request_id

 struct ALMAS_LoadReceiverHierarchy {
  long long request_id;
  string Filename ;};
 #pragma keylist ALMAS_LoadReceiverHierarchy request_id

 struct ALMAS_LoadTemplateSet {
  long long  request_id;
  string Filename; };
 #pragma keylist ALMAS_LoadTemplateSet request_id

 struct ALMAS_LoadConfiguration {
  long long  request_id;
  string Filename; };
 #pragma keylist ALMAS_LoadConfiguration request_id

 struct ALMAS_UpdateDynamicMessageData {
  long long request_id;
  string ProducerID;
  ALMAS_DataModel::ALMAS_AlertIDType AlertID;
  string DataValue;
  ALMAS_DataModel::ALMAS_DynamicMessageDataType OldData; };
 #pragma keylist ALMAS_UpdateDynamicMessageData request_id

 struct ALMAS_SetAlertInhibited {
  long long request_id;
  string ProducerID;
  ALMAS_DataModel::ALMAS_TemplateIDType TemplateID;
  boolean Inhibition; };
 #pragma keylist ALMAS_SetAlertInhibited request_id
```

```
 struct ALMAS_AttachCategorisationRule {
  long long request_id;
  long RuleID;
  ALMAS_DataModel::ALMAS_TemplateIDType TemplateID; };
#pragma keylist ALMAS_AttachCategorisationRule request_id

 struct ALMAS_DetachCategorisationRule {
  long long request_id;
  long RuleID;
  ALMAS_DataModel::ALMAS_TemplateIDType TemplateID; };
#pragma keylist ALMAS_DetachCategorisationRule request_id

 struct ALMAS_RemoveAlertsWithDynamicMessageData  {
  long long request_id;
  string CancellerID;
  string DataType;
  string DataValue; };
 #prgma keylist ALMAS_RemoveAlertsWithDynamicMessageData request_id

};
endif
```

10.3.3 DCPS topics QoS

ALMAS topics share the same values for most of the DDS QoS (cf. [DDS]):

QoS	Value
USER_DATA	<unspecified>
TOPIC_DATA	<unspecified>
GROUP_DATA	<unspecified>
PRESENTATION	<unspecified>
DEADLINE	Period = infinite
LATENCY_BUDGET	duration = <unspecified>
OWNERSHIP	EXCLUSIVE
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:

- 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

10.4 DLRL

10.4.1 ALMAS Client

The ALMAS client module is not required in the DDS PSM since this is all available through the use of the standard DDS mechanisms and the topics already defined for ALMAS_StateType and ALMAS_Alert (i.e., through the DCPS mapping).

10.4.2 ALMAS Management IDL

```
// Copyright 2005-2007 THALES, BAE Systems, Raytheon

#include “dds_dlrl.idl”
#include “ALMAS_DataModel.idl”
#ifndef __ALMAS_Management_DEF
#define __ALMAS_Management_DEF
#pragma prefix "omg.org"

module ALMAS_Management {

 typedef sequence<ALMAS_DataModel::ALMAS_Alert> ALMAS_AlertSet;

 typedef sequence<ALMAS_DataModel::ALMAS_TemplateIDType> ALMAS_TemplateIDTypeSet;
```

```
valuetype ALMAS_Manager : DDS::ObjectRoot {

 attribute string ALMAS_SystemID;

 // alert retrieval methods

 ALMAS_DataModel::ALMAS_CallStatus GetAlert (
  in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
  out ALMAS_DataModel::ALMAS_Alert Alert);

 ALMAS_DataModel::ALMAS_CallStatus GetAlerts (
  in string Filter,
  out ALMAS_AlertSet AlertSet);

 // ALMAS-wide control methods

 ALMAS_DataModel::ALMAS_CallStatus SetAlertInhibited (
  in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
  in boolean Inhibition);

 ALMAS_DataModel::ALMAS_CallStatus UpdateDynamicMessageData (
  in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
  in string DataValue,
  in ALMAS_DataModel::ALMAS_DynamicMessageDataType OldData);

 ALMAS_DataModel::ALMAS_CallStatus RegisterNotificationListener (
  in ALMAS_Client::ALMAS_NotificationListener Handle);

 // Template management methods

 ALMAS_DataModel::ALMAS_CallStatus GetTemplate (
  in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
  out ALMAS_DataModel::ALMAS_AlertTemplateType Template);

 ALMAS_DataModel::ALMAS_CallStatus GetAllTemplateIDs (
  in string Filter,
  out ALMAS_TemplateIDTypeSet TemplateIDSet);
};

valuetype ALMAS_ManagerExtensions : ALMAS_Manager {

 ALMAS_DataModel::ALMAS_CallStatus RemoveAlertsWithDynamicData (
  in string CancellerID,
  in string DataType,
  in string DataValue);
```

```
  ALMAS_DataModel::ALMAS_CallStatus AttachCategorisationRule (
    in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
    in long RuleID);

  ALMAS_DataModel::ALMAS_CallStatus DetachCategorisationRule (
    in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
    in long RuleID);
 };

valuetype ALMAS_Producer : DDS::ObjectRoot {

  ALMAS_DataModel::ALMAS_CallStatus RaiseAlertFromOverrides (
    in string ProducerID,
    in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
    in ALMAS_DataModel::ALMAS_AlertDataAttributesType Attributes,
    in boolean CategoryValid,
    in boolean PriorityValid,
    in boolean StatusValid,
    in boolean ScopeValid,
    in boolean TimeoutValid,
    in boolean ConfirmationRequiredValid,
    in boolean SecondaryGroupingValid,
    in boolean PersistentValid,
    in boolean ReliablyDistributedValid,
    in boolean TimeoutActionValid,
    in boolean AcknowledgementModelValid,
    in boolean StaticMessagesValid,
    in boolean DynamicMessagesValid,
    out ALMAS_DataModel::ALMAS_AlertIDType AlertID);

  ALMAS_DataModel::ALMAS_CallStatus RaiseAlertFromData (
    in string ProducerID,
    in ALMAS_DataModel::ALMAS_AlertTemplateType AlertInfo,
    out ALMAS_DataModel::ALMAS_AlertIDType AlertID);

  ALMAS_DataModel::ALMAS_CallStatus RaiseAlertFromTemplate (
    in string ProducerID,
    in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
    out ALMAS_DataModel::ALMAS_AlertIDType AlertID);

  ALMAS_DataModel::ALMAS_CallStatus UpdateAlertPriority (
    in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
    in string ProducerID,
    in short Priority);

  ALMAS_DataModel::ALMAS_CallStatus CancelAlert (
```

```
    in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
    in string CancellerID,
    in string CancellationReason);
};

valuetype ALMAS_Responder : DDS::ObjectRoot {

 ALMAS_DataModel::ALMAS_CallStatus RegisterReceiver (
    in ALMAS_Client::ALMAS_Receiver Handle,
    in string ReceiverID,
    in string RKType);

 ALMAS_DataModel::ALMAS_CallStatus UnregisterReceiver (
    in string ReceiverID);

 ALMAS_DataModel::ALMAS_CallStatus AcknowledgeAlert (
    in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
    in string ReceiverID);

 ALMAS_DataModel::ALMAS_CallStatus HandleAlert (
    in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
    in string ReceiverID);

 ALMAS_DataModel::ALMAS_CallStatus ConfirmReceipt (
    in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
    in string ReceiverID);
};

valuetype ALMAS_ResponderExtensions : ALMAS_Responder {

 ALMAS_DataModel::ALMAS_CallStatus SetLanguage (
    in string ReceiverID,
    in string Language);

 ALMAS_DataModel::ALMAS_CallStatus GetFilledMessageText (
    in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
    out ALMAS_DataModel::ALMAS_StringSet Messages);
};

valuetype ALMAS_Configuration : DDS::ObjectRoot {

 ALMAS_DataModel::ALMAS_CallStatus LoadReceiverHierarchy (
    in string Filename );

 ALMAS_DataModel::ALMAS_CallStatus LoadTemplateSet (
    in string Filename );
```

```
  ALMAS_DataModel::ALMAS_CallStatus LoadConfiguration (
    in string Filename );
 };
};

#endif
```

11 COM IDL Platform Specific Model

11.1 Rationale

The objective of this PSM is to normalize the structures and interfaces required for a COM implementation of the standard. This PSM aims to support the entire PIM interface.

In order for this interface to be reasonably compatible with the other PSMs provided in this document, the data model part is separated from the functional interface part.

All attributes, methods, and associations are mapped to COM IDL elements. As a general rule, therefore, classes with methods are mapped to COM interfaces, classes without methods are mapped to structs, attributes are mapped to interface read/write methods. All return parameters and exceptions are mapped to method out parameters with the COM HRESULT returned from all interface methods.

Subscribe methods and indication classes are also mapped within a client IDL file that has to be implemented by clients in order to receive indications (i.e., callbacks) from ALMAS.

11.2 ALMAS Data Model IDL

```
// Copyright 2005-2007 THALES, BAE Systems, Raytheon
import "oaidl.idl";
import "ocidl.idl";

#ifndef __ALMAS_DataModel_DEF
#define __ALMAS_DataModel_DEF

 typedef long ALMAS_AlertIDType;

 typedef long ALMAS_TemplateIDType;

 typedef long ALMAS_TimeoutType;

#ifdef NOLONGLONG
 typedef struct {
  unsigned long low;
  unsigned long high;
 } ALMAS_DateTimeType;
#else
 typedef unsigned long long ALMAS_DateTimeType; // long long to be EVoT compatible
#endif


typedef enum {
   ALMAS_Action = 1,
   ALMAS_Warning,
```

```
  ALMAS_Information,
  ALMAS_Situation} ALMAS_CategoryType;

typedef enum {
  ALMAS_Raised = 1,
  ALMAS_Routed,
  ALMAS_Received,
  ALMAS_Acknowledged,
  ALMAS_Handled,
  ALMAS_Cancelled,
  ALMAS_TimedOut} ALMAS_StateType;

typedef enum {
  ALMAS_Actual = 1,
  ALMAS_Exercise,
  ALMAS_System,
  ALMAS_Test} ALMAS_StatusType;

typedef enum {
  ALMAS_PublicScope = 1,
  ALMAS_RestrictedScope,
  ALMAS_PrivateScope} ALMAS_ScopeType;

typedef enum {
  ALMAS_CancelOnly = 1,
  ALMAS_NotifyOnly,
  ALMAS_CancelWithNotify} ALMAS_TimeoutActionType;

typedef enum {
  ALMAS_AckByNone = 1,
  ALMAS_AckByAnyone,
  ALMAS_AckByAll} ALMAS_AckModelType;

 typedef struct {
  boolean Success;
  short Reason;
  BSTR Description;} ALMAS_CallStatus;

 typedef struct {
  SAFEARRAY(BSTR) AlternativeAction;
  short ActioneePriority; } ALMAS_ValidAlertResponseType;

 typedef [uuid(0B7DF643-8DFF-4cfe-BC48-3C2E07BD6A79)]struct ALMAS_ReceiverKindType {
  BSTR RKType;
  BSTR RKParentType;
  ALMAS_ValidAlertResponseType ValidResponse; } ALMAS_ReceiverKindType;
```

```
  typedef [uuid(62FD9C37-ED08-46b2-8122-8B783D83DC5E)] struct ALMAS_DynamicMessageDataType{
  BSTR DataType;
  BSTR DataTag;
  BSTR DataValue; } ALMAS_DynamicMessageDataType;


typedef [uuid(06A4B73D-52AD-4009-BC0A-4FC940D3A799)]struct ALMAS_StaticMessageType{
  BSTR MessageText;
  BSTR MessageLanguage; } ALMAS_StaticMessageType;


typedef [uuid(F42A96DE-F513-4880-8E5A-5C2B308A2898)]struct ALMAS_AlertDataExtraAttributesType{
  BSTR Name;
  short TypeOfByteData;
  BSTR Description;
   SAFEARRAY(byte) Value; } ALMAS_AlertDataExtraAttributesType;


typedef struct {
  ALMAS_TemplateIDType TemplateID;
  ALMAS_CategoryType Category;
  short Priority;
  ALMAS_StatusType Status;
  ALMAS_ScopeType Scope;
  ALMAS_TimeoutType Timeout;
  boolean ConfirmationRequired;
  BSTR SecondaryGrouping;
  boolean Persistent;
  boolean ReliablyDistributed;
  ALMAS_TimeoutActionType TimeoutAction;
  ALMAS_AckModelType AcknowledgementModel;
  SAFEARRAY(ALMAS_StaticMessageType) StaticMessages;
  SAFEARRAY(ALMAS_DynamicMessageDataType)DynamicMessages;
  SAFEARRAY(ALMAS_AlertDataExtraAttributesType) ExtraAttributes;} ALMAS_AlertDataType;

typedef struct {
  boolean Inhibited;
  boolean RaiseToAll;
  ALMAS_AlertDataType AlertData;
  SAFEARRAY(ALMAS_ReceiverKindType) ReceiverKinds; } ALMAS_AlertTemplateType;

typedef struct {
  boolean Acknowledged;
  boolean Routed;
  boolean Actioned;
```

```
  boolean ReceiverIsActionee;
  SAFEARRAY(BSTR) AlternativeAction;
  BSTR ReceiverID;
  ALMAS_AlertIDType AlertID; } ALMAS_AlertReportType;

 typedef struct {
  BSTR ReceiverID;
  ALMAS_ReceiverKindType ReceiverKind; } ALMAS_AvailableAlertReceiverType;


 typedef struct {
  ALMAS_AlertIDType AlertID;
  ALMAS_DateTimeType RaisingTime;
  ALMAS_StateType CurrentState;
  BSTR ProducerID;
  ALMAS_AlertDataType AlertData;
  SAFEARRAY(ALMAS_AvailableAlertReceiverType) Receivers; } ALMAS_Alert;
#endif
```

11.3 ALMAS Client IDL

```
// Copyright 2005-2008 THALES, BAE Systems, Raytheon

import "../Alert_Data_Router/ALMAS_DataModel.idl";
#ifndef __ALMAS_Client_DEF
#define __ALMAS_Client_DEF

[object,uuid(13D0EBD4-47C0-4661-BFF6-B8220219BD66),pointer_default(unique)]
 interface IALMAS_Receiver: IUnknown {

  HRESULT StateChangeNotification  (
    [n]  ALMAS_AlertIDType AlertID,
    [in]  ALMAS_StateType NewState);

  HRESULT AlertDataNotification  (// alert D is embedded within info
    [in]  ALMAS_Alert AlertInfo,
    [in] ALMAS_AlertReportType *Report); // changed to in in.
 };

 [object,uuid(2BA3B7FA-40EB-4021-8828-36243C457379),pointer_default(unique)]
 interface IALMAS_NotificationListener: IUnknown {

  HRESULT AlertDistributionNotification  (
    [in] ALMAS_AlertIDType AlertID);

  HRESULT Get_ALMAS_SystemIDNotification (
```

```
    [in] BSTR * ALMAS_SystemID);

  HRESULT GetAlertNotification(
    [in] ALMAS_Alert Alert);

  HRESULT GetAlertsNotification(
    [in] SAFEARRAY(ALMAS_Alert)AlertSet);

  HRESULT GetTemplateNotification(
    [in] ALMAS_AlertTemplateType AlertTemplate); // Corrected to return the template, not the Template ID.

  HRESULT GetTemplatesNotification(
    [in] SAFEARRAY(ALMAS_TemplateIDType) TemplateIDSet);

 };
#endi
```

11.4 ALMAS Management IDL

```
// Copyright 2005-2008 THALES, BAE Systems, Raytheon

import "../Alert_Data_Router/ALMAS_Client.idl";
import "../Alert_Data_Router/ALMAS_DataModel.idl";
#ifndef __ALMAS_Management_DEF
#define __ALMAS_Management_DEF


// typedef struct {
//   unsigned long MaxSize;
//   unsigned long LengthUsed;
//   [size_is(MaxSize), length_is(LengthUsed), unique] ALMAS_TemplateIDType *pValue;}
ALMAS_TemplateIDTypeSet;

 [object,uuid(3BC17616-F798-421A-8FB9-DDC0A8259CE3),pointer_default(unique)]
 interface IALMAS_Manager : IUnknown {

  HRESULT Get_ALMAS_SystemID(IALMAS_NotificationListener *Handle);

  // alert retrieval methods

  HRESULT GetAlert (
    [in]  ALMAS_AlertIDType AlertID,
    [in]  IALMAS_NotificationListener *Handle,
    [out] ALMAS_CallStatus *CallStatus);

  HRESULT GetAlerts (
```

```
    [in]  BSTR Filter,
    [in]  IALMAS_NotificationListener *Handle,
    [out] ALMAS_CallStatus *CallStatus);

  // ALMAS-wide control methods

  HRESULT SetAlertInhibited (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  ALMAS_TemplateIDType TemplateID,
    [in]  boolean Inhibition);

  HRESULT UpdateDynamicMessageData (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  ALMAS_AlertIDType AlertID,
    [in]  BSTR ObjectValue,
    [in]  ALMAS_DynamicMessageDataType OldValue);

  HRESULT RegisterNotificationListener (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  IALMAS_NotificationListener *Handle);

  // Template management methods

  HRESULT GetTemplate (
    [in]  IALMAS_NotificationListener *Handle,
    [in]  ALMAS_TemplateIDType TemplateID,
    [out] ALMAS_CallStatus *CallStatus);

  HRESULT GetAllTemplateIDs (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  BSTR Filter,
    [in]  IALMAS_NotificationListener *Handle);
};

[object,uuid(6AE3866D-3EF5-4BBD-B2ED-261DBCFF2307),pointer_default(unique)]
interface IALMAS_ManagerExtensions : IALMAS_Manager {

  HRESULT RemoveAlertsWithDynamicData (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  BSTR CancellerID,
    [in]  BSTR DataType,
    [in]  BSTR DataValue);

  HRESULT AttachCategorisationRule (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  ALMAS_TemplateIDType TemplateID,
```

```
    [in]  long RuleID);

  HRESULT DetachCategorisationRule (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  ALMAS_TemplateIDType TemplateID,
    [in]  long RuleID);
 };

 [object,uuid(32033A16-EC76-4AC5-A457-D607B5CFD0CF),pointer_default(unique)]
 interface IALMAS_Producer : IUnknown {
// SDG Changed optional parameters to pointers
  HRESULT RaiseAlertFromOverrides (
    [out] ALMAS_AlertIDType *AlertID,
    [in]  BSTR ProducerID,
    [in]  ALMAS_TemplateIDType TemplateID,
    [in]  ALMAS_CategoryType Category,
    [in] boolean ValidCategory
    [in] short Priority,
    [in] boolean ValidPriority,
    [in] ALMAS_StatusType AlertStatus,
    [in] boolean ValidStatus,
    [in] ALMAS_ScopeType Scope,
    [in] boolean ValidScope,
    [in] ALMAS_TimeoutType Timeout,
    [in] boolean ValidTimeout,
    [in] boolean ConfirmationRequired,
    [in] boolean ValidConfirmationRequired,
    [in] BSTR SecondaryGrouping,
    [in] boolean ValidSecondaryGrouping,
    [in] boolean Persistent,
    [in] boolean ValidPersistent,
    [in] boolean ReliablyDistributed,
    [in] boolean ValidReliablyDistributed,
    [in[ ALMAS_TimeoutActionType TimeoutAction,
    [in] boolean ValidTimeoutAction,
    [in] ALMAS_AckModelType AcknowledgementModel,
    [in] boolean ValidAcknowledgementModel,
    [in] SAFEARRAY(ALMAS_StaticMessageType) StaticMessages,
    [in] boolean ValidStaticMessages,
    [in] SAFEARRAY(ALMAS_DynamicMessageDataType) DynamicMessageData,
    [in] boolean ValidDynamicMessageData,
    [in] SAFEARRAY(ALMAS_ReceiverKindType) AlertReceivers,
    [in] boolean ValidAlertReceiverSet,
    [out] ALMAS_CallStatus *CallStatus);

  HRESULT RaiseAlertFromData (
```

```
    [out] ALMAS_AlertIDType *AlertID,
    [in]  BSTR ProducerID,
    [in] ALMAS_AlertTemplateType AlertInfo,
    [out] ALMAS_CallStatus *CallStatus);

  HRESULT RaiseAlertFromTemplate (
    [out] ALMAS_AlertIDType *AlertID,
    [in]  BSTR ProducerID,
    [in]  ALMAS_TemplateIDType TemplateID,
    [out] ALMAS_CallStatus *CallStatus);

  HRESULT UpdateAlertPriority (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  ALMAS_AlertIDType AlertID,
    [in]  BSTR ProducerID,
    [in]  short Priority);

  HRESULT CancelAlert (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  ALMAS_AlertIDType AlertID,
    [in]  BSTR CancellerID,
    [in]  BSTR CancellationReason);
};

[object,uuid(BA617DFD-6DBD-4F08-ACD5-E7F489A113E5),pointer_default(unique)]
interface IALMAS_Responder : IUnknown {

  HRESULT RegisterReceiver (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  IALMAS_Receiver *ReceiverHandle,
    [in]  BSTR ReceiverID,
    [in]  BSTR RKType);

  HRESULT UnregisterReceiver (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  BSTR ReceiverID);

  HRESULT AcknowledgeAlert (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  ALMAS_AlertIDType AlertID,
    [in]  BSTR ReceiverID);

  HRESULT HandleAlert (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  ALMAS_AlertIDType AlertID,
    [in]  BSTR ReceiverID);
```

```
 HRESULT ConfirmReceipt (
   [out] ALMAS_CallStatus *CallStatus,
   [in]  ALMAS_AlertIDType AlertID,
   [in]  BSTR ReceiverID);
};

[object,uuid(CC748587-4926-45D7-B52E-4A88000A3426),pointer_default(unique)]
interface IALMAS_ResponderExtensions : IALMAS_Responder {

 HRESULT SetLanguage (
   [out] ALMAS_CallStatus *CallStatus,
   [in]  BSTR ReceiverID,
   [in]  BSTR Language);

 HRESULT GetFilledMessageText (
   [out] ALMAS_CallStatus *CallStatus,
   [in]  ALMAS_AlertIDType AlertID,
   [in] [out] BSTR MessageText);
};

[object,uuid(C3B50C13-8124-4A5F-98B8-9C68D9D1BDE9),pointer_default(unique)]
interface IALMAS_Configuration : IUnknown {

 HRESULT LoadReceiverHierarchy (
   [out] ALMAS_CallStatus *CallStatus,
   [in]  BSTR Filename);

 HRESULT LoadTemplateSet (
   [out] ALMAS_CallStatus *CallStatus,
   [in]  BSTR Filename);

 HRESULT LoadConfiguration (
   [out] ALMAS_CallStatus *CallStatus,
   [in]  BSTR Filename);
};
#endif
```

INDEX