April 2008

ALert MAnagement Service (ALMAS) Specification

Beta 1

OMG Document Number: dtc/2008-04-01

Standard document URL: http://www.omg.org/spec/ALMAS/1.0/PDF

Associated File(s)*: http://www.omg.org/spec/ALMAS/20080201

http://www.omg.org/spec/ALMAS/20080202

- original files: c4i/2008-02-03 (PIM XMI), c4i/2008-02-04 (PSMs)

This OMG document replaces the submission document (c4i/2008-02-01, Alpha). It is an OMG Adopted Beta Specification and is currently in the finalization phase. Comments on the content of this document are welcome, and should be directed to issues@omg.org by November 1, 2008.

You may view the pending issues for this specification from the OMG revision issues web page *http://www.omg.org/issues/*.

The FTF Recommendation and Report for this specification will be published on February 23, 2009. If you are reading this after that date, please download the available specification from the OMG Specifications Catalog.

USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

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

LICENSES

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

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

PATENTS

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

GENERAL USE RESTRICTIONS

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

DISCLAIMER OF WARRANTY

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

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

RESTRICTED RIGHTS LEGEND

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

TRADEMARKS

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

COMPLIANCE

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

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

OMG's Issue Reporting Procedure

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

Table of Contents

Preface

About the Object Management Group

OMG

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

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

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

OMG Specifications

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

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

Specifications within the Catalog are organized by the following categories:

OMG Modeling Specifications

UML

MOF

XMI

CWM

Profile specifications

OMG Middleware Specifications

CORBA/IIOP

IDL/Language Mappings

Specialized CORBA specifications

CORBA Component Model (CCM).

Platform Specific Model and Interface Specifications

CORBAservices

CORBAfacilities

OMG Domain specifications

OMG Embedded Intelligence specifications

OMG Security specifications.

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

OMG Headquarters

140 Kendrick Street

Building A, Suite 300

Needham, MA 02494

USA

Tel: +1-781-444-0404

Fax: +1-781-444-0320

Email: *pubs@omg.org*

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

Typographical Conventions

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

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

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

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

Helvetica/Arial - 10 pt: Exceptions

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

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

This specification introduces two main ‘levels’ of conformance.

- Conformance level 0 is PIM level conformance. To achieve level 0 an implementation must demonstrably conform to the PIM as presented in sections 6.1 to 6.3 of this document (i.e., data structures and interfaces that match the PIM). In addition the PIM-PSM mappings used by the vendor should be presented and be consistently applied. The implementation should also conform to the XML Alert template data model, and the XML initialization PSMs as presented in section 7.1 to 7.3 of this document.
- Conformance level 1 is PSM level conformance. An ALMAS achieving conformance level 1 conforms to one or more of the middleware platform specific models presented in Chapters 8, 9, and 10 of this document in addition to conforming to the XML Alert template data model and the XML initialization PSMs as presented in sections 7.1 to 7.3 of this document.

In addition any class postfixed with 'Extensions' and the categorization model PIM in section 6.4 and PSM in section 7.4 are providing optional functionality above and beyond the core requirements of an ALMAS. Any implementation should state which, if any, of these it implements.

Effectively these levels of conformance allow for ALMAS implementations utilizing any middleware technology to conform (at level 0) to the standard in a consistent manner, but still allowing for ALMAS implementations using middleware specified in Chapters 8, 9, and 10, and therefore at conformance level 1, to be truly inter-operable.

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 a 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

The following set of standard terminology will be used within this document.

- 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 actions.
- **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 which 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, there is an expectation that the ALMAS specification will include 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

[1] The definition of the required action is not within the scope of ALMAS.

6 Platform Independent Model (PIM)

The PIM has been split into four packages as follows:

- ALMAS Client Callbacks
- ALMAS Data Model
- ALMAS Management
- ALMAS Categorization

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

6.1 ALMAS Client Callbacks

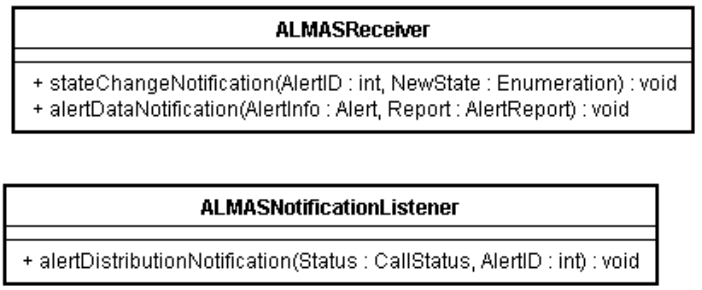

Figure 6.1 - PIM class diagram for ALMAS Clients

This package contains a definition of the interfaces which the ALMAS system expects to be supported by ALMAS clients.

6.1.1 ALMASNotificationListener

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

Operation

Name	Type	Summary
AlertDistributionNotification (CallStatus, int)	public void [Parameters] Status: CallStatus, AlertID: int	This is called as soon as a safety critical alert has been received by the ALMAS manager.

6.1.2 ALMASReceiver

Class provided by registering alert receivers for provision of the notification callbacks.

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.
alertDataNotification (Alert, AlertReport)	public void [Parameters] AlertInfo: Alert, Report: AlertReport	Provides notifications of new and modified alert data.

6.2 ALMAS Data Model

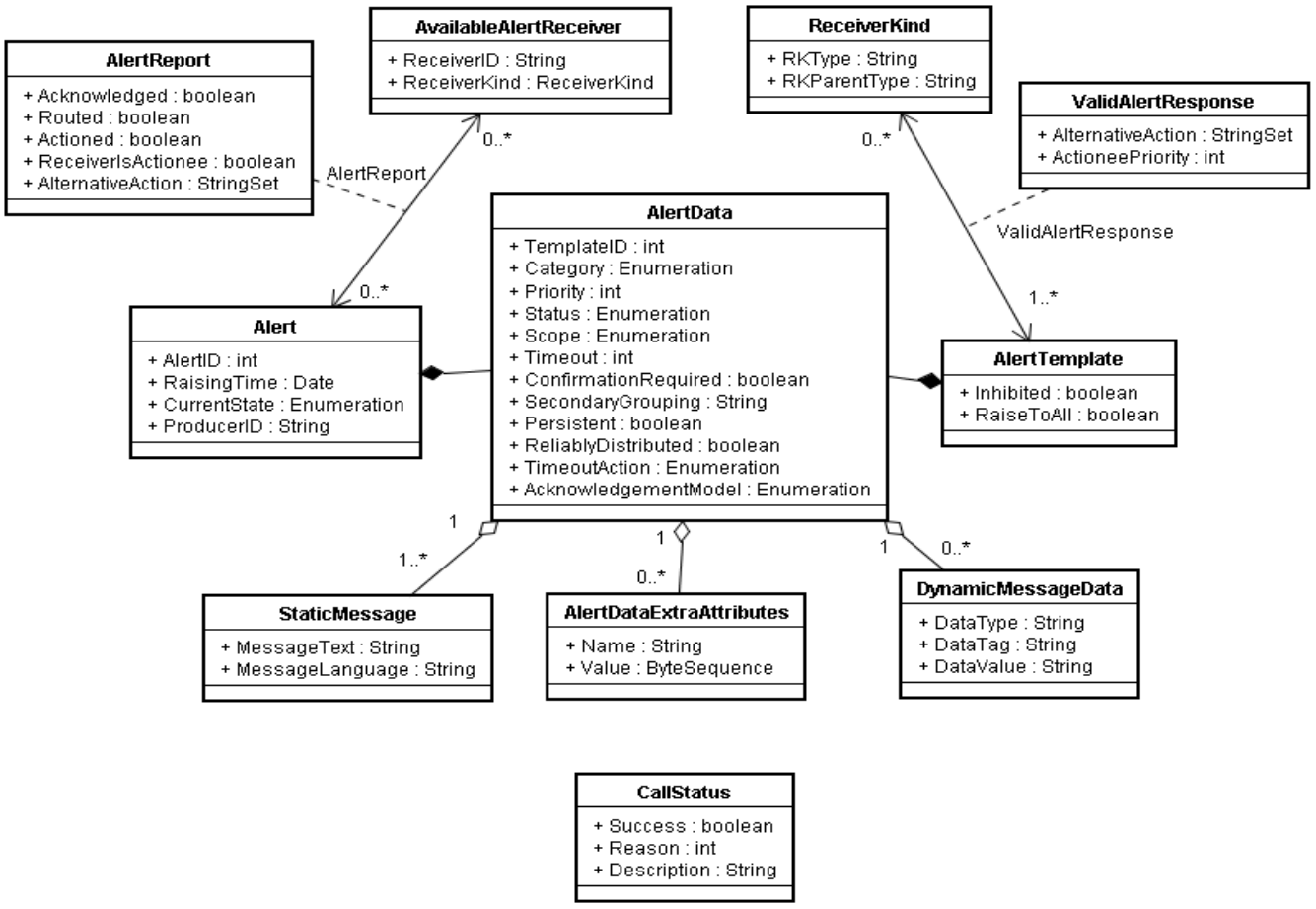

Figure 6.2 - PIM class diagram for ALMAS Data Model

This package contains the data model used by the ALMAS API packages.

6.2.1 Alert

An active alert within ALMAS. The message text is not fully filled in here.

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, Canceled}
ProducerID	public String	The producer freetext ID - corresponds to CAP source

6.2.2 AlertData

This represents the set of data shared between the alert template and alert classes. All fields have default values which 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.

Attribute

Name	Type	Summary
TemplateID	public int	A unique identifier for template which 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
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.

Name	Type	Summary
Scope	public Enumeration	Corresponds to CAP scope.
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 which 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}

6.2.3 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 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 configuration xml PSM specified in section 7.2.

Attribute

Name	Type	Summary
Name	public String	Name of the client specific attribute
Value	public ByteSequence	Contents as a byte sequence

6.2.4 AlertReport

This provides the status information for specifically delivered alert item to a receiver. This will contain details of whether the instance has been acknowledged by this receiver etc. and will also be completed with respect to any dynamic message data.

This distributes a copy of the data to the receiver.

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.

6.2.5 AlertTemplate

A Template specifying the generic characteristics of a specific alert type (e.g., the general characteristics of a collision warning alert). This includes the category of alert, such as Action, etc.

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.

6.2.6 AvailableAlertReceiver

A registered receiver of alerts.

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.

6.2.7 CallStatus

This is the ALMAS success/failure descriptor class. If Success, then the other parameters are not applicable.

Reason =

0 = Success

1 = Not Accepted

2 = Malformed Alert

3 = Timeout/delivery

4 = Service Unavailable

5+ = Other

Description provides a string definition of the problem, particularly in event of Reason = Other.

Attribute

Name	Type	Summary
Success	public boolean	
Reason	public int	
Description	public String	

6.2.8 DynamicMessageData

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.

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 position of the related object in the alert template text e.g., alert text is "xxxxx %t1 yyyyyyy %t2 zzzz" and related objects have tags of 't1' and 't2' and values of 123 and 456 resulting in a alert of 'xxxxx 123 yyyyyyy 456 zzzz' (ignores locale / syntax of the text string issues)
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.

6.2.9 ReceiverKind

The descriptor of an alert receiver. For example, this could be an operator role.

Attribute

Name	Type	Summary
RKType	public String	String identifier of the kind of receiver, for example the role of a receiving operator. This may also include special subcategories such as 'Raise to all.'
RKParentType	public String	

6.2.10 StaticMessage

Provides the message text as a tuplet with the language in which the text is provided.

Attribute

Name	Type	Summary
MessageText	Public String	In an alert report this attribute is the final resolved message. In the template this is the default message.
MessageLanguage	public String	The message ‘Locale’

6.2.11 ValidAlertResponse

Class defining the valid behavior of alerts based on a specific template for a specific kind of receiver.

Attribute

Name	Type	Summary
AlternativeAction	public StringSet	The 'names' of alternative actions to be presented to the relevant actor.
ActioneePriority	public int	The priority of the receiver kind as actionee for a specifc alert kind 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 class.

6.3 ALMAS Management

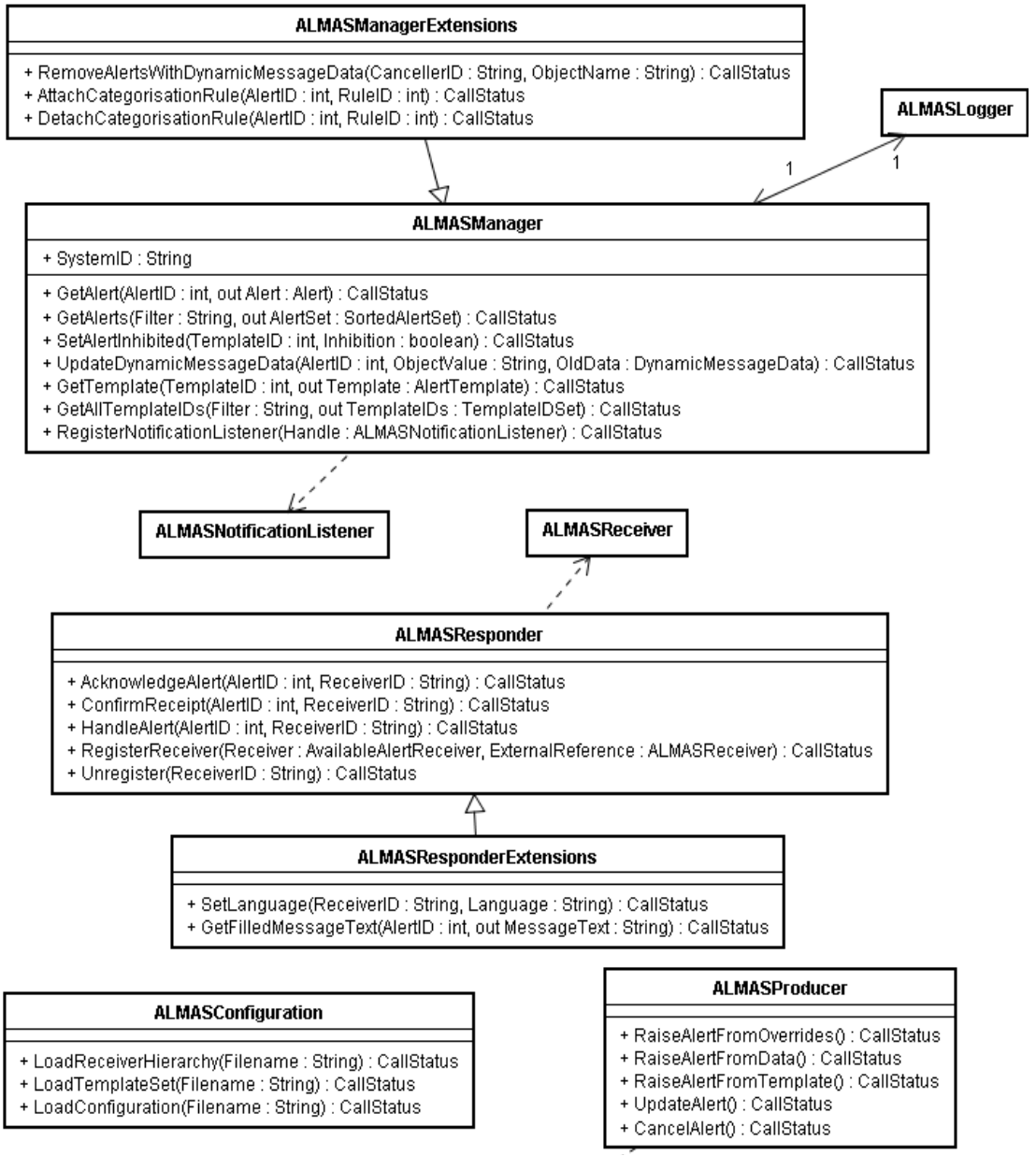

Figure 6.3 - PIM class diagram for ALMAS Management

(Note ALMASProducer method parameters are omitted for clarity)

This package provides the main API to the ALMAS service.

6.3.1 ALMASConfiguration

Provides an API by which clients can configure ALMAS to behave in a more tailored manner in order to satisfy very specific requirements.

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.
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.
LoadConfiguration (String)	public CallStatus [Parameters] Filename: String	Loads the ALMAS configuration file as provided by the client.

6.3.2 ALMASLogger

Logging mechanism to record historical Alert information.

6.3.3 ALMASManager

The controller for alert entities. Primary entry point for the API. Note that the registration of receivers is done via the ALMAS Responder class.

Attribute

Name	Type	Summary
SystemID	private String	Corresponds to CAP sender

Operation

Name	Type	Summary
GetAlert (int, Alert)	public CallStatus [Parameters] AlertID: int, out Alert: Alert	Retrieves data for a specific raised alert from ALMAS
GetAlerts (String, SortedAlertSet)	public CallStatus [Parameters] Filter: String, out AlertSet: SortedAlertSet	Retrieves a set of all alert instances within ALMAS that satisfy the filter.

Name	Type	Summary
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: DynamicMessageData	Indicates a change to the value of a related object for the provided alert ID.
GetTemplate (int, AlertTemplate)	public CallStatus [Parameters] TemplateID: int, out Template: AlertTemplate	Retrieves an existing alert template from ALMAS.
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.
RegisterNotificationListener (ALMASNotificationListener)	public CallStatus [Parameters] Handle: ALMASNotificationListener	Registers a new Notification Listener for receipt of the alert distribution notifications.

6.3.4 ALMASManagerExtensions

This class contains optional extensions to the alert manager functionality.

Operation

Name	Type	Summary
RemoveAlertsWithDynamic MessageData (String, String)	public CallStatus [Parameters] CancellerID: String, ObjectName: 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. ObjectName corresponds to RelatedObject Value parameter. Implementation is optional.
AttachCategorizationRule (int, int)	public CallStatus [Parameters] AlertID: int, RuleID: int	Associates a categorization rule with an alert.

DetachCategorizationRule (int, int)	public CallStatus [Parameters] AlertID: int, RuleID: int	Disassociates a categorization rule from an alert.

6.3.5 ALMASProducer

Provides the API by which a client producing alerts can create and manipulate the alerts that they own.

Operation

Name	Type	Summary
RaiseAlertFromOverrides (String, int, int, String, Enumeration, Enumeration, Enumeration, Enumeration, double, boolean, StringSet, StringSet, int, Enumeration, Enumeration)	public CallStatus [Parameters] ProducerID: String, TemplateID: int, out AlertID: int, Message: String, MessageLanguage: Enumeration, Category: Enumeration, Status: Enumeration, Scope: Enumeration, Timeout: double, Confirmationrequired: boolean, DynamicMessageDataSet: StringSet, AlertReceiverSet: StringSet, Priority: int, TimeoutAction: Enumeration, AcknowledgementModel: Enumeration	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, AlertData, int)	public CallStatus [Parameters] ProducerID: String, Alert: AlertData, out AlertID: int	Raise an alert not present in the ALMAS template database 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.

UpdateAlert (int, String, AlertData)	public CallStatus [Parameters] AlertID: int, ProducerID: String, Alert: AlertData	Update an existing raised alert instance within ALMAS.
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.

6.3.6 ALMASResponder

Provides the API for alert receivers, including registration and responses.

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.
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 (AvailableAlertReceiver, ALMASReceiver)	public CallStatus [Parameters] Receiver: AvailableAlertReceiver, ExternalReference: ALMASReceiver	Registers a new receiver in ALMAS.
Unregister (String)	public CallStatus [Parameters] ReceiverID: String	Removes a registered receiver from ALMAS, indicating that they are no longer avail-able for receipt of alert data.

6.3.7 ALMASResponderExtensions

Optional extensions to the alert responder functionality.

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.

6.4 Alert Categorization (optional PIM)

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.

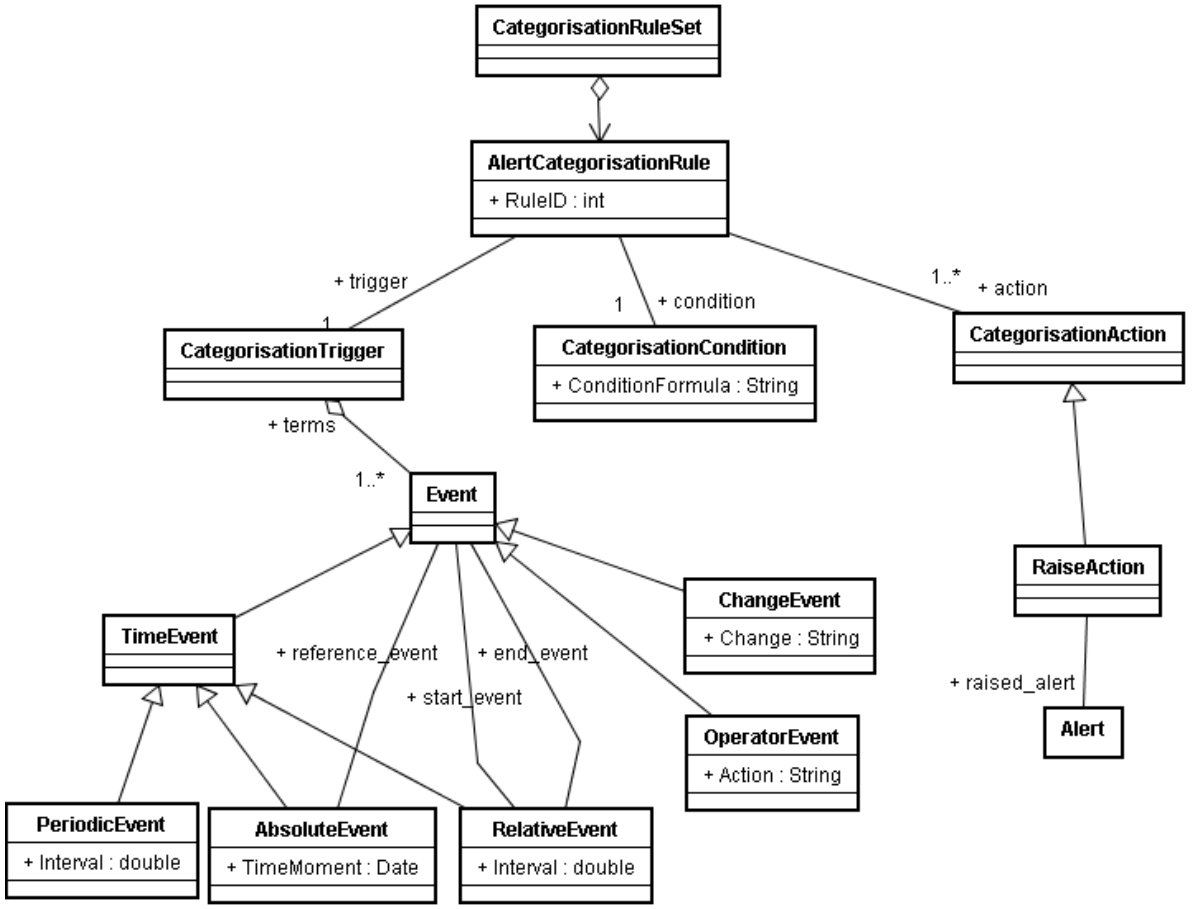

Figure 6.4 - Alert Categorization Platform Independent Model

6.4.1 AbsoluteEvent

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

Attribute

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

6.4.2 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.

Attribute

Name	Type	Summary
RuleID	public int	The rule identifier

6.4.3 CategorizationAction

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

6.4.4 CategorizationCondition

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

Attribute

Name	Type	Summary
ConditionFormula	public String	The condition formula

6.4.5 CategorizationRuleSet

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

6.4.6 CategorizationTrigger

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

6.4.7 ChangeEvent

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

Attribute

Name	Type	Summary
Change	public String	The change that is required.

6.4.8 Event

General class of Event, used within the Categorization Trigger.

6.4.9 OperatorEvent

Operator initiated events, for example operator changing a role.

Attribute

Name	Type	Summary
Action	public String	The operator action required.

6.4.10 PeriodicEvent

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

Attribute

Name	Type	Summary
Interval	public double	The condition formula

6.4.11 RaiseAction

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

6.4.12 RelativeEvent

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

Attribute

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

6.4.13 Time Event

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

6.5 Dynamic Behavior

6.5.1 Action Situation Alert State Model

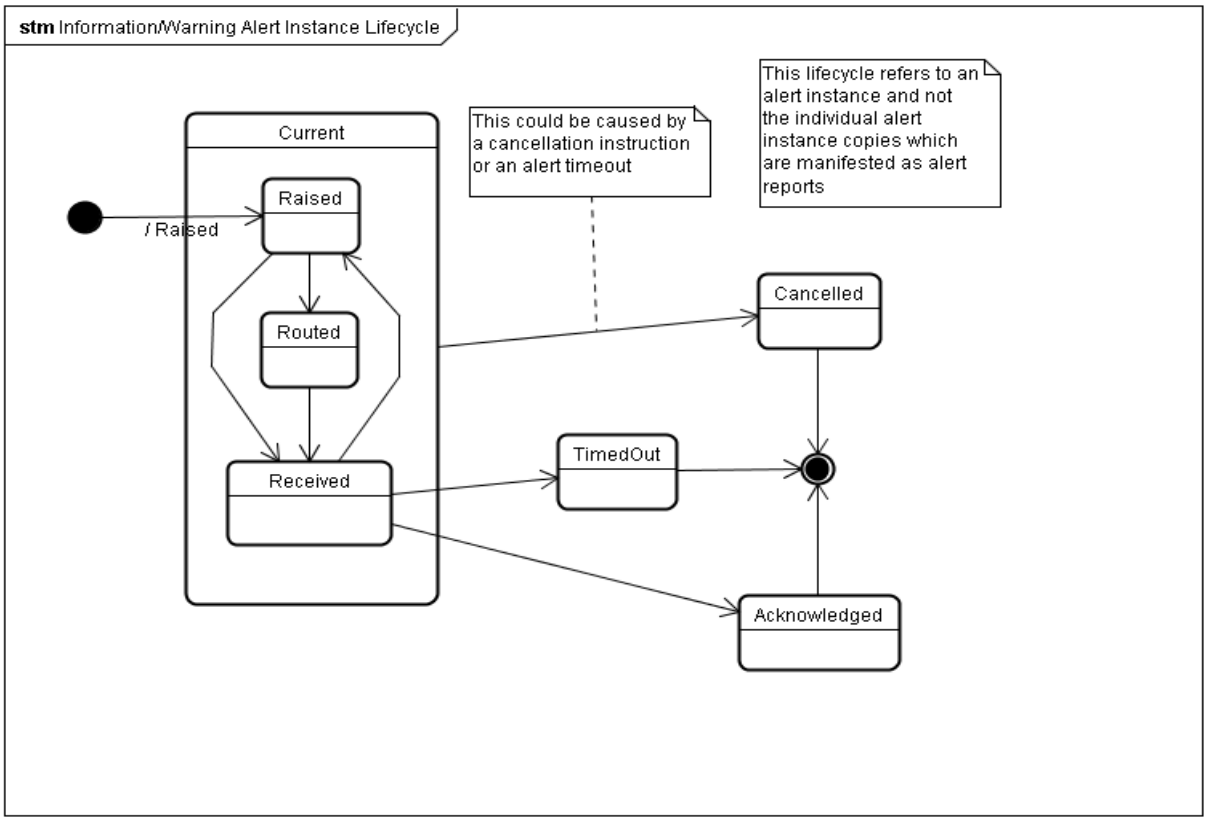

Figure 6.5 - Action/Situation Alert Lifecycle

6.5.2 Information Warning Alert State Model

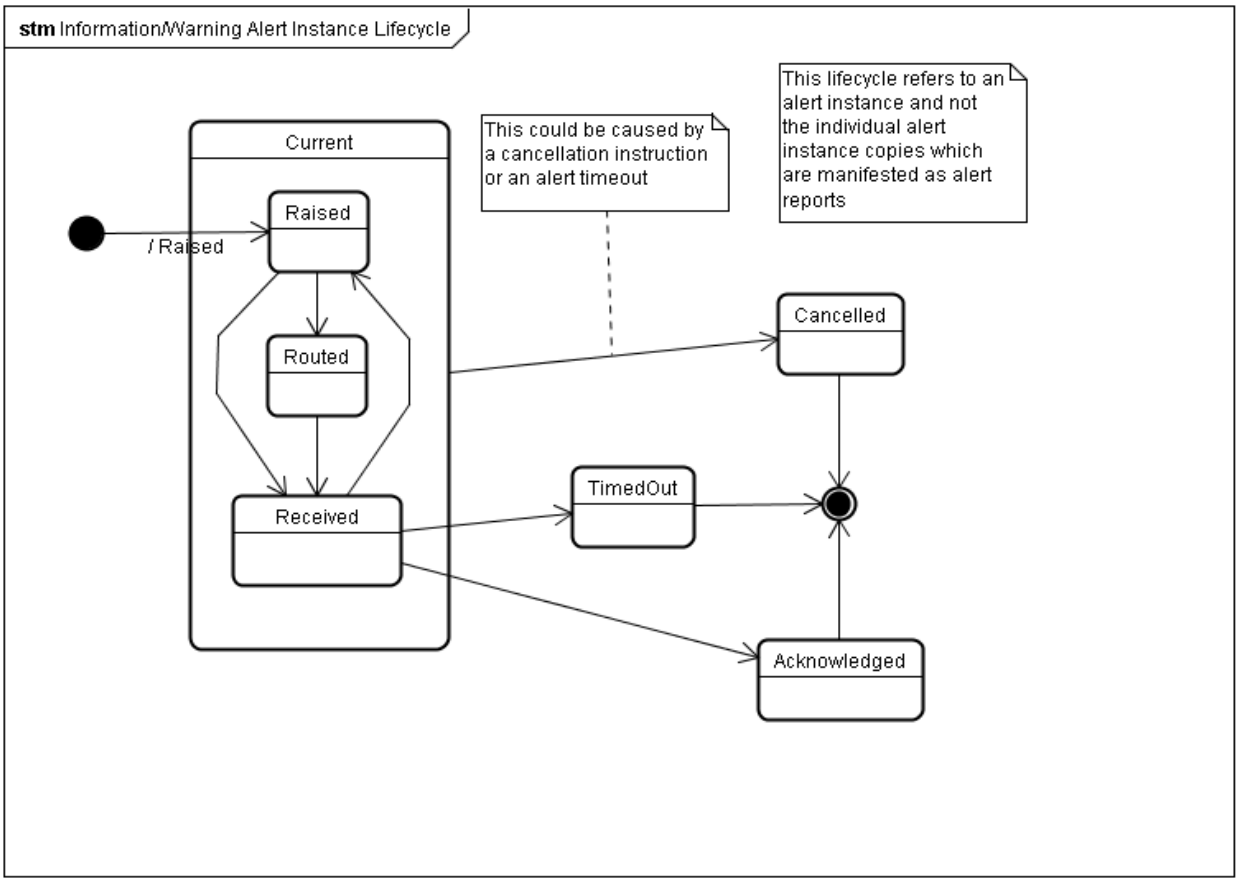

Figure 6.6 - Information/Warning Alert Instance Lifecycle

6.5.3 Alert Registration and Creation

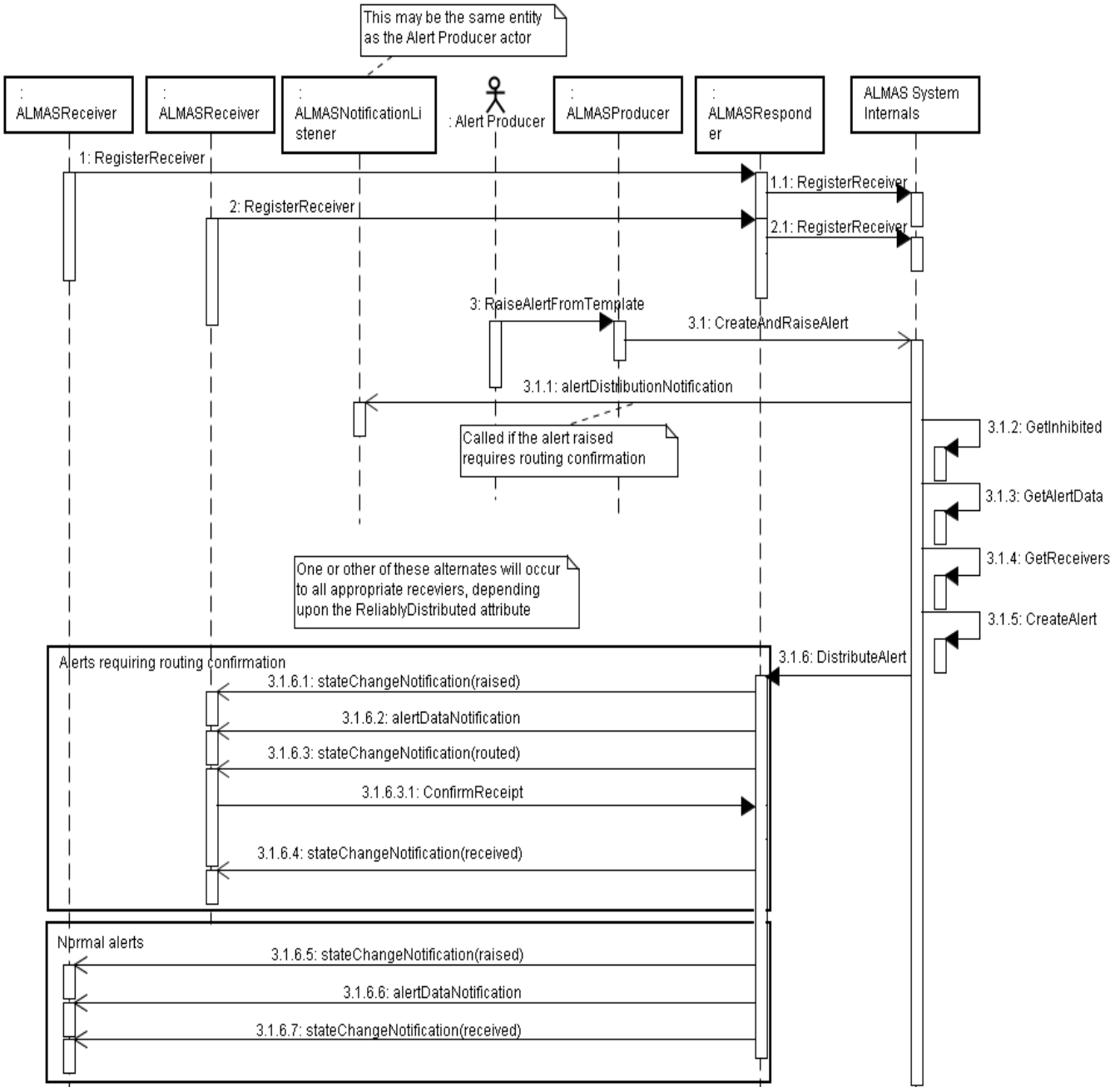

Figure 6.7 -Alert Registration and Creation Sequence Diagram

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.

7 XML Platform Specific Model

7.1 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" attributeFormDefault="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" maxOccurs="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"
         minOccurs="1" maxOccurs="10"/>
```

```
      <xs:element name="Alert_Data_Extra_Attributes"
               type="Alert_Data_Extra_Attributes_T" minOccurs="0"
               maxOccurs="10"/>
      <xs:element name="Dynamic_Message_Data"
               type="Dynamic_Message_Data_T" minOccurs="0"
               maxOccurs="10"/>
      <xs:element name="Alert_Routing" type="Alert_Routing_T"
                minOccurs="0" maxOccurs="30"/>
   </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: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: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:maxLength value="20"/>
            </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:integer">
               <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:maxLength value="10"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:element>
         <xs:element name="Alert_Responses" minOccurs="0"
                  maxOccurs="10">
            <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:maxLength value="10"/>
               </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>
```

7.2 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>
```

7.3 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"-->
        <xs: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="50">
                <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:maxLength value="10" />
                </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:maxLength value="10" />
                </xs:restriction>
            </xs:simpleType>
        </xs:element>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

7.4 The ALMAS Categorization Rule File (Optional PSM)

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="Categorization Rule Set" type="Categorization Rule Set"/>
    <xs:complexType name="Categorization Rule Set">
        <xs:sequence>
            <xs:element name="Alert Categorization Rule" type="Alert
                    Categorization Rule"/>
```

```
   </xs:sequence>
</xs:complexType>
<xs:element name="Alert Categorization Rule" type="Alert Categorization
    Rule"/>
<xs:complexType name="Alert Categorization Rule">
   <xs:sequence>
      <xs:element name="ruleID" type="xs:int"/>
      <xs:element name="action" type="Categorization Action"
                maxOccurs="unbounded"/>
      <xs:element name="condition" type="Categorization
                Condition"/>
      <xs:element name="trigger" type="Categorization Trigger"/>
   </xs:sequence>
</xs:complexType>
<xs:element name="Categorization Trigger" type="Categorization
     Trigger"/>
<xs:complexType name="Categorization 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="Time Event">
         <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="Time Event">
         <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="Time Event">
         <xs:sequence>
            <xs:element name="time_moment" type="xs:date"/>
         </xs:sequence>
      </xs:extension>
   </xs:complexContent>
</xs:complexType>
<xs:element name="Categorization Action" type="Categorization Action"/>
<xs:complexType name="Categorization Action">
   <xs:sequence/>
</xs:complexType>
<xs:element name="Categorization Condition" type="Categorization
     Condition"/>
<xs:complexType name="Categorization 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="Categorization 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="Time Event" type="Time Event"/>
    <xs:complexType name="Time Event">
        <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>
```

8 OMG CORBA/IDL Platform Specific Model

8.1 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 which deal with errors through CORBA exceptions.

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

8.2 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,
  Canceled};

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_AlertDataExtraAttributesType {
  string Name;
  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;
   string 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
```

8.3 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,
    in ALMAS_DataModel::ALMAS_CallStatus Status);
 };

};

#endif
```

8.4 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 ALMAS_DataModel::ALMAS_DynamicMessageDataType ObjectInfo);

 ALMAS_DataModel::ALMAS_CallStatus AttachCategorisationRule (
   in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
   in long RuleID);
```

```
 ALMAS_DataModel::ALMAS_CallStatus DetachCategorisationRule (
   in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
   in long RuleID);
};

interface ALMAS_Producer {

 ALMAS_DataModel::ALMAS_CallStatus RaiseAlertFromOverrides (
   in string ProducerID,
   in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
   in string Message,
   in string MessageLanguage,
   in ALMAS_DataModel::ALMAS_CategoryType Category,
   in ALMAS_DataModel::ALMAS_StatusType Status,
   in ALMAS_DataModel::ALMAS_ScopeType Scope,
   in ALMAS_DataModel::ALMAS_TimeoutType Timeout,
   in boolean ConfirmationRequired,
   in ALMAS_DataModel::ALMAS_DynamicMessageDataTypeSet DynamicMessageData,
   in ALMAS_DataModel::ALMAS_AvailableAlertReceiverTypeSet AlertReceivers,
   in short Priority,
   in ALMAS_DataModel::ALMAS_TimeoutActionType TimeoutAction,
   in ALMAS_DataModel::ALMAS_AckModelType AcknowledgementModel,
   out ALMAS_DataModel::ALMAS_AlertIDType AlertID);

 ALMAS_DataModel::ALMAS_CallStatus RaiseAlertFromData (
   in string ProducerID,
   in ALMAS_DataModel::ALMAS_AlertDataType AlertData,
   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 UpdateAlert (
   in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
   in string ProducerID,
   in ALMAS_DataModel::ALMAS_AlertDataType AlertData);

 ALMAS_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 Handle,
   in ALMAS_DataModel::ALMAS_AvailableAlertReceiverType Receiver);

 ALMAS_DataModel::ALMAS_CallStatus Unregister (
   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);
};

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
```

9 DDS Platform Specific Model

9.1 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 the following when compared to the CORBA PSM:
    - use of valuetypes instead of interfaces – note that a valuetype which is to be distributed by DLRL must inherit from DDS::ObjectRoot
    - There must be an XML-based mapping from DLRL to CDPS. 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.

9.1.1 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, 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 which 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 modelled 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 not 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.

9.2 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 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};

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_AlertDataExtraAttributesType {
  string Name;
  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; };
  #pragma keylist ALMAS_AlertTemplateType AlertData.TemplateID

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

9.3 DCPS

9.3.1 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`.

9.3.2 ALMAS Management

The following table provides explanation of the mapping of methods in the ALMAS Management module. Only those methods which are mapped directly to DDS level constructs are listed in the table, all methods which 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	UpdateDynamicMessagData (String, String, DynamicMessageData)	DDS write of a new instance of Alert with a new value
ALMAS Manager	GetTemplate (int)	DDS read with query condition.
ALMAS Manager	SetAlertInhibited (int, boolean)	DDS write of a new instance of Alert with a new value
ALMAS Manager	RegisterNotificationListener (ALMAS Notification Listener)	Creation of a new DDS Listener.
ALMAS Manager Extensions	RemoveAlertsWithDynamicMessageData (String, String)	DDS dispose with condition
ALMAS Producer	RaiseAlertFromOverrides (String, int, int, String, Enumeration, Enumeration, Enumeration, Enumeration, double, boolean, StringSet, StringSet, int, Enumeration, Enumeration)	DDS read w/condition of AlertTemplate topic followed by write of Alert topic
ALMAS Producer	RaiseAlertFromData (String, Alert Data, int)	DDS write of alert topic
ALMAS Producer	updateAlert (int, String, Alert Data)	DDS read w/condition followed by write of Alert topic
ALMAS Producer	cancelAlert (int, String, String)	DDS read w/condition followed by write of new instance of Alert with currentState modified
ALMAS Responder	RegisterReceiver (Available Alert Receiver)	DDS creation of a new listener/data reader

ALMAS Responder	Unregister (String)	DDS removal of listener/data reader

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

#ifndef __ALMAS_DataModel_DEF
#define __ALMAS_DataModel_DEF

#include "ALMAS_DataModel.idl"

module ALMAS_Management {

 typedef sequence<ALMAS_DataModel::ALMAS_Alert> ALMAS_AlertSet;

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

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

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

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

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

 struct ALMAS_HandleAlert {
   long request_id;
   ALMAS_DataModel::ALMAS_AlertIDType AlertID;
```

```
  string ReceiverID;};
#pragma keylist ALMAS_HandleAlert

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

struct ALMAS_SetLanguage {
  long request_id;
  string ReceiverID;
  string Language;};
#pragma keylist ALMAS_SetLanguage

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

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

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

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

struct ALMAS_AttachCategorisationRule {
  long request_id;
  long RuleID;
  ALMAS_DataModel::ALMAS_AlertIDType AlertID; };
#pragma keylist ALMAS_AttachCategorisationRule
```

```
 struct ALMAS_DetachCategorisationRule {
   long request_id;
   long RuleID;
   ALMAS_DataModel::ALMAS_AlertIDType AlertID; };
 #pragma keylist ALMAS_DetachCategorisationRule

};
#endif
```

9.3.3 DCPS topics QoS

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

Table 10.1 - QoS of DCPS topics

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

9.4 DLRL

9.4.1 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).

9.4.2 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 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);
};

valuetype ALMAS_ManagerExtensions : ALMAS_Manager {

 ALMAS_DataModel::ALMAS_CallStatus RemoveAlertsWithDynamicData (
   in string CancellerID,
```

```
    in ALMAS_DataModel::ALMAS_DynamicMessageDataType ObjectInfo);

  ALMAS_DataModel::ALMAS_CallStatus AttachCategorisationRule (
    in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
    in long RuleID);

  ALMAS_DataModel::ALMAS_CallStatus DetachCategorisationRule (
    in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
    in long RuleID);
 };

valuetype ALMAS_Producer : DDS::ObjectRoot {

  ALMAS_DataModel::ALMAS_CallStatus RaiseAlertFromOverrides (
    in string ProducerID,
    in ALMAS_DataModel::ALMAS_TemplateIDType TemplateID,
    in string Message,
    in string MessageLanguage,
    in ALMAS_DataModel::ALMAS_CategoryType Category,
    in ALMAS_DataModel::ALMAS_StatusType Status,
    in ALMAS_DataModel::ALMAS_ScopeType Scope,
    in ALMAS_DataModel::ALMAS_TimeoutType Timeout,
    in boolean ConfirmationRequired,
    in ALMAS_DataModel::ALMAS_DynamicMessageDataTypeSet DynamicMessageData,
    in ALMAS_DataModel::ALMAS_AvailableAlertReceiverTypeSet AlertReceivers,
    in short Priority,
    in ALMAS_DataModel::ALMAS_TimeoutActionType TimeoutAction,
    in ALMAS_DataModel::ALMAS_AckModelType AcknowledgementModel,
    out ALMAS_DataModel::ALMAS_AlertIDType AlertID);

  ALMAS_DataModel::ALMAS_CallStatus RaiseAlertFromData (
    in string ProducerID,
    in ALMAS_DataModel::ALMAS_AlertDataType AlertData,
    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 UpdateAlert (
   in ALMAS_DataModel::ALMAS_AlertIDType AlertID,
   in string ProducerID,
   in ALMAS_DataModel::ALMAS_AlertDataType AlertData);

 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 ALMAS_DataModel::ALMAS_AvailableAlertReceiverType Receiver);

 ALMAS_DataModel::ALMAS_CallStatus Unregister (
   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 string MessageText);
  };

  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
```

10 COM IDL Platform Specific Model

10.1 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 which has to be implemented by clients in order to receive indications (i.e., callbacks) from ALMAS.

10.2 ALMAS Data Model IDL

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

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

#ifndef __ALMAS_DataModel_DEF
#define __ALMAS_DataModel_DEF

 typedef long ALMAS_AlertIDType;

 typedef long ALMAS_TemplateIDType;

 typedef long ALMAS_TimeoutType;

 typedef long long ALMAS_DateTimeType; // long long to be EVoT compatible

 typedef struct {
   unsigned long MaxSize;
```

```
  unsigned long LengthUsed;
  [size_is(MaxSize), length_is(LengthUsed), unique] byte *pValue;} ALMAS_ByteSequence;

typedef struct {
  unsigned long MaxSize;
  unsigned long LengthUsed;
  [size_is(MaxSize), length_is(LengthUsed), unique] LPSTR *pValue;} ALMAS_StringSet;

typedef enum {
  Action = 1,
  Warning,
  Information,
  Situation} ALMAS_CategoryType;

typedef enum {
  Raised = 1,
  Routed,
  Received,
  Acknowledged,
  Handled,
  Cancelled} ALMAS_StateType;

typedef enum {
  Actual = 1,
  Exercise,
  System,
  Test} ALMAS_StatusType;

typedef enum {
  PublicScope = 1,
  RestrictedScope,
  PrivateScope} ALMAS_ScopeType;

typedef enum {
  CancelOnly = 1,
  NotifyOnly,
  CancelWithNotify} ALMAS_TimeoutActionType;

typedef enum {
  AckByNone = 1,
```

```
    AckByAnyone,
    AckByAll} ALMAS_AckModelType;

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

  typedef struct {
    LPSTR AlternativeAction;
    short ActioneePriority; } ALMAS_ValidAlertResponseType;

  typedef struct {
    LPSTR RKType;
    LPSTR RKParentType;
    ALMAS_ValidAlertResponseType ValidResponse; } ALMAS_ReceiverKindType;

  typedef struct {
    unsigned long MaxSize;
    unsigned long LengthUsed;
    [size_is(MaxSize), length_is(LengthUsed), unique] ALMAS_ReceiverKindType *pValue;}
ALMAS_ReceiverKindTypeSet;

  typedef struct {
    LPSTR DataType;
    LPSTR DataTag;
    LPSTR DataValue; } ALMAS_DynamicMessageDataType;

  typedef struct {
    unsigned long MaxSize;
    unsigned long LengthUsed;
    [size_is(MaxSize), length_is(LengthUsed), unique] ALMAS_DynamicMessageDataType *pValue;}
ALMAS_DynamicMessageDataTypeSet;

  typedef struct {
    LPSTR MessageText;
    LPSTR MessageLanguage; } ALMAS_StaticMessageType;

  typedef struct {
    unsigned long MaxSize;
```

```
   unsigned long LengthUsed;
   [size_is(MaxSize), length_is(LengthUsed), unique] ALMAS_StaticMessageType *pValue;}
ALMAS_StaticMessageTypeSet;

 typedef struct {
   LPSTR Name;
   ALMAS_ByteSequence Value; } ALMAS_AlertDataExtraAttributesType;

 typedef struct {
   unsigned long MaxSize;
   unsigned long LengthUsed;
   [size_is(MaxSize), length_is(LengthUsed), unique] ALMAS_AlertDataExtraAttributesType
*pValue;} ALMAS_AlertDataExtraAttributesTypeSet;

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

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

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

```
    boolean Actioned;
    boolean ReceiverIsActionee;
    LPSTR AlternativeAction;
    LPSTR ReceiverID;
    ALMAS_AlertIDType AlertID; } ALMAS_AlertReportType;

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

  typedef struct {
    unsigned long MaxSize;
    unsigned long LengthUsed;
    [size_is(MaxSize), length_is(LengthUsed), unique] ALMAS_AvailableAlertReceiverType *pValue;
} ALMAS_AvailableAlertReceiverTypeSet;

  typedef struct {
    ALMAS_AlertIDType AlertID;
    ALMAS_DateTimeType RaisingTime;
    ALMAS_StateType CurrentState;
    LPSTR ProducerID;
    ALMAS_AlertDataType AlertData;
    ALMAS_AvailableAlertReceiverTypeSet Receivers; } ALMAS_Alert;
#endif
```

10.3 ALMAS Client IDL

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

import “ALMAS_DataModel.idl”
#ifndef __ALMAS_Client_DEF
#define __ALMAS_Client_DEF

 [object,uuid(...),pointer_default(unique)]
 interface IALMAS_Receiver: IUnknown {

  HRESULT ALMAS_Client_StateChangeNotification (
    [in]  ALMAS_AlertIDType AlertID,
    [in]  ALMAS_StateType NewState);
```

```
  HRESULT ALMAS_Client_AlertDataNotification (// alert ID is embedded within info
    [in]  ALMAS_Alert AlertInfo,
    [out] ALMAS_AlertReportType *Report);
 };

 [object,uuid(...),pointer_default(unique)]
 interface IALMAS_NotificationListener: IUnknown {

  HRESULT AlertDistributionNotification (
    [in]  ALMAS_AlertIDType AlertID,
    [in]  ALMAS_CallStatus Status);
 };
#endif
```

10.4 ALMAS Management IDL

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

import “ALMAS_Client.idl”
import “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_Alert *pValue;} ALMAS_AlertSet;

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

 [object,uuid(...),pointer_default(unique)]
 interface IALMAS_Manager : IUnknown {

  HRESULT _get_ALMAS_SystemID ([out] LPSTR * ALMAS_SystemID);
  HRESULT _put_ALMAS_SystemID ([in]  LPSTR * ALMAS_SystemID);

  // alert retrieval methods

  HRESULT GetAlert (
```

```
    [in]  ALMAS_AlertIDType AlertID,
    [out] ALMAS_Alert *Alert,
    [out] ALMAS_CallStatus *Status);

  HRESULT GetAlerts (
    [in]  LPSTR Filter,
    [out] ALMAS_AlertSet *AlertSet,
    [out] ALMAS_CallStatus *Status);

  // ALMAS-wide control methods

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

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

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

  // Template management methods

  HRESULT GetTemplate (
    [out] ALMAS_AlertTemplateType *AlertTeplate,
    [in]  ALMAS_TemplateIDType TemplateID,
    [out] ALMAS_CallStatus *Status);

  HRESULT GetAllTemplateIDs (
    [out] ALMAS_CallStatus *Status,
    [in]  LPSTR Filter,
    [out] ALMAS_TemplateIDTypeSet *TemplateIDSet);
};

[object,uuid(...),pointer_default(unique)]
interface IALMAS_ManagerExtensions : IALMAS_Manager {

  HRESULT RemoveAlertsWithDynamicData (
    [out] ALMAS_CallStatus *CallStatus,
```

```
    [in]  LPSTR CancellerID,
    [in]  ALMAS_DynamicMessageDataType ObjectInfo);

  HRESULT AttachCategorisationRule (
    [out] ALMAS_CallStatus *Status,
    [in]  ALMAS_AlertIDType AlertID,
    [in]  long RuleID);

  HRESULT DetachCategorisationRule (
    [out] ALMAS_CallStatus *Status,
    [in]  ALMAS_AlertIDType AlertID,
    [in]  long RuleID);
};

[object,uuid(...),pointer_default(unique)]
interface IALMAS_Producer : IUnknown {

  HRESULT RaiseAlertFromOverrides (
    [out] ALMAS_AlertIDType *AlertID,
    [in]  LPSTR ProducerID,
    [in]  ALMAS_TemplateIDType TemplateID,
    [in]  LPSTR Message,
    [in]  LPSTR MessageLanguage,
    [in]  ALMAS_CategoryType Category,
    [in]  ALMAS_StatusType AlertStatus,
    [in]  ALMAS_ScopeType Scope,
    [in]  ALMAS_TimeoutType Timeout,
    [in]  boolean ConfirmationRequired,
    [in]  ALMAS_DynamicMessageDataTypeSet DynamicMessageData,
    [in]  ALMAS_ReceiverKindTypeSet AlertReceivers,
    [in]  short Priority,
    [in]  ALMAS_TimeoutActionType TimeoutAction,
    [in]  ALMAS_AckModelType AcknowledgementModel,
    [out] ALMAS_CallStatus *Status);

  HRESULT RaiseAlertFromData (
    [out] ALMAS_AlertIDType *AlertID,
    [in]  LPSTR ProducerID,
    [in]  ALMAS_AlertDataType AlertData,
    [out] ALMAS_CallStatus *Status);

  HRESULT RaiseAlertFromTemplate (
    [out] ALMAS_AlertIDType *AlertID,
    [in]  LPSTR ProducerID,
```

```
    [in]  ALMAS_TemplateIDType TemplateID,
    [out] ALMAS_CallStatus *Status);

  HRESULT UpdateAlert (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  ALMAS_AlertIDType AlertID,
    [in]  LPSTR ProducerID,
    [in]  ALMAS_AlertDataType AlertData);

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

[object,uuid(...),pointer_default(unique)]
interface IALMAS_Responder : IUnknown {

  HRESULT RegisterReceiver (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  IALMAS_Receiver *Handle,
    [in]  ALMAS_AvailableAlertReceiverType Receiver);

  HRESULT Unregister (
    [out] ALMAS_CallStatus *CallStatus,
    [in]  LPSTR ReceiverID);

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

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

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

```
 [object,uuid(...),pointer_default(unique)]
 interface IALMAS_ResponderExtensions : IALMAS_Responder {

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

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

 [object,uuid(...),pointer_default(unique)]
 interface IALMAS_Configuration : IUnknown {

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

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

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

11 Changes or Extensions Required to Adopted OMG Specifications

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