Technical Specification Group Services and System Aspects
Meeting #17, Biarritz, France, 9-12 September 2002

TSGS#17(02)0477

Source: **SA5 (Telecom Management)**

Title: **Rel-5 CRs 32.111-1/2/3 (Fault Management; Alarm Integration Reference Point)**

Document for: **Approval**

Agenda Item: **7.5.3**

Doc-1st-	Spec	CR	Rev	Phase	Subject	Cat	Version	Doc-	Workite	Relation
SP-020477	32.111-1	004	-	Rel-5	**Add requirements for new clearAlarms() operation in Alarm IRP**	B	5.0.0	S5-026491	OAM-NIM	**Parent CR**
SP-020477	32.111-2	017	-	Rel-5	**Add clearAlarms() operation for Alarm IRP:IS**	B	5.0.0	S5-026490	OAM-NIM	**Child CR**
SP-020477	32.111-3	018	-	Rel-5	**Add clearAlarm and other updates**	B	5.0.0	S5-026492	OAM-NIM	**Grandchild CR**

3GPP TSG-SA5 (Telecom Management) **S5-026491**
Meeting #29, Beijing, PRC, June 2002

CR-Form-v5

CHANGE REQUEST

⌘ **32.111-1** **CR** **004** ⌘ **rev** **-** ⌘ Current version: **5.0.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE ☐ Radio Access Network **X** Core Network **X**

Title:	⌘	Add requirements for new clearAlarms() operation in Alarm IRP
Source:	⌘	SA5
Work item code:	⌘	OAM-NIM
Date:	⌘	28/06/2002
Category:	⌘	**B**
Release:	⌘	*REL-5*

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(addition of feature),*
C *(functional modification of feature)*
D *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
REL-4 (Release 4)
REL-5 (Release 5)

Reason for change:	⌘	Align requirements with addition of a new operation to support the agreed Automatic Detection and Manual Clear (ADMC) feature.
Summary of change:	⌘	Add the new clearAlarms() operation.
Consequences if not approved:	⌘	The ADMC requirement will not be satisfied.

Clauses affected:	⌘	4.1.3, 5.2.3		
Other specs affected:	⌘	☐ Other core specifications ☐ Test specifications **X** O&M Specifications	⌘	 32.111-2, 32.111-3, 32.111-4
Other comments:	⌘	If this "parent" CR is approved, then also its related "child" CR32.111-2CR017 (S5-026490) and its "grand-child" CR32.111-3CR020 (S5-026492) can be approved.		

4.1.3 Clearing of alarms

The alarms originated in consequence of faults need to be cleared. To clear an alarm it is necessary to repair the corresponding fault. The procedures to repair faults are implementation dependent and therefore they are out of the scope of the present document, however, in general:

- the equipment faults are repaired by replacing the faulty units with working ones;
- the software faults are repaired by means of partial or global system initialisations, by means of software patches or by means of updated software loads;
- the communication faults are repaired by replacing the faulty transmission equipment or, in case of excessive noise, by removing the cause of the noise;
- the QOS faults are repaired either by removing the causes that degraded the QOS or by improving the capability of the system to react against the causes that could result in a degradation of the QOS;
- Solving the environmental problem repairs the environment faults (high temperature, high humidity, etc.).

It is also possible that an ADAC fault is spontaneously repaired, without the intervention of the operator (e.g. a threshold crossed fault). In this case the NE behaves as for the ADAC faults repaired by the operator.

In principle, the NE uses the same mechanisms to detect that a fault has been repaired, as for the detection of the occurrence of the fault. However, for ADMC faults, manual intervention by the operator is always necessary to clear the fault. Practically, various methods exist for the system to detect that a fault has been repaired and clear alarms and the faults that triggered them. For example:

- The system operator implicitly requests the NE to clear a fault, e.g. by initialising a new device that replaces a faulty one. Once the new device has been successfully put into service, the NE shall clear the fault(s). Consequently, the NE shall clear all related alarms.
- The system operator explicitly requests the clearing of one or more alarms. Once the alarm(s) has/have been cleared, ~~the NE shall detect that the fault condition has ceased~~the fault management system (within EMS and/or NE) should reissue those alarms (as new alarms) in case the fault situation still persists.
- The NE detects the exchange of a faulty device by a new one and initialises it autonomously. Once the new device has been successfully put into service, the NE shall clear the fault(s). Consequently, the NE shall clear all related alarms.
- The NE detects that a previously reported threshold crossed alarm is no longer valid. It shall then clear the corresponding active alarm and the associated fault, without requiring any operator intervention. The details for the administration of thresholds and the exact condition for the NE to clear a threshold crossed alarm are implementation specific and depend on the definition of the threshold measurement, see also subclause 4.1.1.
- ADMC faults/alarms can, by definition, not be cleared by the NE autonomously. Therefore, in any case, system operator functions shall be available to request the clearing of ADAC alarms/faults in the NE. Once an ADMC alarm/fault has been cleared, the NE shall clear the associated ADAC fault/alarm.

Details of these mechanisms are system/implementation specific.

Each time an alarm is cleared the NE shall generate an appropriate clear alarm event. A clear alarm is defined as an alarm, as specified in subclause 3.1, except that its severity is set to "cleared". The relationship between the clear alarm and the active alarm is established:

- by re-using a set of parameters that uniquely identify the active alarm (cf. subclause 4.1.1); or
- by including a reference to the active alarm in the clear alarm.

When a clear alarm is generated the corresponding active alarm is removed from the active alarm list.

. . .

5.2.3 Alarm clearing

On the Itf-N, alarm reports containing the value "cleared" of the parameter perceivedSeverity are used to clear the alarms. The correlation between the clear alarm and the related active alarms is performed by means of unambiguous identifiers.

This clearing mechanism ensures the correct clearing of alarms, independently of the (manufacturer-specific) implementation of the mapping of alarms/state change events in accordance with the information model of the Itf-N.

The IRP manager may also clear alarms manually.

3GPP TSG-SA5 (Telecom Management) **S5-026490**
Meeting #29, Beijing, PRC, June 2002

CR-Form-v5

CHANGE REQUEST

⌘ **32.111-2** **CR** **017** ⌘ **rev** **-** ⌘ Current version: **5.0.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE ☐ Radio Access Network **X** Core Network **X**

Title:	⌘	Add clearAlarms() operation for Alarm IRP:IS
Source:	⌘	SA5
Work item code:	⌘	OAM-NIM
Date:	⌘	28/06/2002
Category:	⌘	**B**
Release:	⌘	*REL-5*

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(addition of feature),*
C *(functional modification of feature)*
D *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
REL-4 (Release 4)
REL-5 (Release 5)

Reason for change:	⌘	Add a new operation to support the agreed Automatic Detection and Manual Clear (ADMC) feature.
Summary of change:	⌘	Add the new clearAlarms() operation.
Consequences if not approved:	⌘	The ADMC requirement will not be satisfied.

Clauses affected:	⌘	5.2.1, 5.3.1.2, 5.4.1.2, 5.5.1, 6.1, 6.3.1.2, (new) 6.7, (old) 6.7.2.1, (old) 6.7.2.2, (old) 6.7.3.1, (old) 6.7.3.2, (old) 6.7.3.3		
Other specs affected:	⌘	☐ Other core specifications ☐ Test specifications **X** O&M Specifications	⌘	 32.111-3, 32.111-4
Other comments:	⌘	This "child" CR can only be approved if its "parent" 32.111-1CR004 (S5-026491) is approved. If this "parent" CR is approved, then also its related "child" CR32.111-3CR020 (S5-026492) can be approved.		

Introduction

The present document is part of a set of TSs which describe the requirements and information model necessary for the Telecommunication Management (TM) of 3G systems. The TM principles and TM architecture are specified in 3GPP TS 32.101 [6] and 3GPP TS 32.102 [7].

A 3G system is composed of a multitude of Network Elements (NE) of various types and, typically, different vendors inter-operate in a co-ordinated manner in order to satisfy the network users' communication requirements.
The occurrence of failures in a NE may cause a deterioration of this NE's function and/or service quality and will, in severe cases, lead to the complete unavailability of the NE. In order to minimise the effects of such failures on the Quality Of Service (QOS) as perceived by the network users it is necessary to:

- detect failures in the network as soon as they occur and alert the operating personnel as fast as possible;
- isolate the failures (autonomously or through operator intervention), i.e. switch off faulty units and, if applicable, limit the effect of the failure as much as possible by reconfiguration of the faulty NE/adjacent NEs;
- if necessary, determine the cause of the failure using diagnosis and test routines; and,
- repair/eliminate failures in due time through the application of maintenance procedures.

This aspect of the management environment is termed "Fault Management" (FM). The purpose of FM is to detect failures as soon as they occur and to limit their effects on the network Quality of Service (QOS) as far as possible. The latter is achieved by bringing additional/redundant equipment into operation, reconfiguring existing equipment/NEs, or by repairing/eliminating the cause of the failure.

Fault Management (FM) encompasses all of the above functionalities except commissioning/decommissioning of NEs and potential operator triggered reconfiguration (these are a matter of Configuration Management (CM), see [13]).

FM also includes associated features in the Operations System (OS), such as the administration of alarm list, the presentation of operational state information of physical and logical devices/resources/functions, and the provision and analysis of the alarm and state history of the network.

1 Scope

The present document defines the Alarm Integration Reference Point (IRP) Information Service (IS), which addresses the alarm surveillance aspects of Fault Management (FM), applied to the N Interface.

The purpose of the AlarmIRP is to define an interface through which a “system” (typically a Network Element Manager or a Network Element) can communicate alarm information for its managed objects to one or several Manager Systems (typically Network Management Systems).

The Alarm IRP IS defines the semantics of alarms and the interactions visible across the reference point in a protocol neutral way. It defines the semantics of the operations and notifications visible in the IRP. It does not define the syntax or encoding of the operations, notifications and their parameters.

2 References

The following documents contain provisions, which through reference in this text constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document.*

[1] ITU-T Recommendation Q821: “Stage 2 and Stage 3 description for the Q3 interface – Alarm surveillance”.

[2] ITU-T Recommendation X.733 (02/92): “Information technology - Open Systems Interconnection - Systems management: Alarm Reporting Function”.

[3] ITU-T Recommendation X.721: “Information Technology - Open Systems Interconnection - Structure Of Management Information: Definition Of Management Information”.

[4] GSM 12.11 version 6.2.0 Release 1997: “Fault management of the Base Station System (BSS)”.

[5] 3GPP TS 32.302: “Notification IRP: Information Service”.

[6] 3GPP TS 32.101: “3G Telecom Management principles and high level requirements”.

[7] 3GPP TS 32.102: “3G Telecom Management architecture”.

[8] 3GPP TS 32.300: “Name Convention for Managed Objects”.

[9] 3GPP TS 32.111-1: “3G Fault Management”.

[10] 3GPP TS 32.622: “Generic Network Resources IRP : Network Resource Model”.

[11] ITU-T Recommendation M.3100 (07/95): “Generic network information model”.

[12] ITU-T Recommendation X.720: “Management Information Model”.

[13] Void.

[14] 3GPP TS 32.312 : “Generic IRP Management : Information Service”.

[15] ITU-T Recommendation X.736: "Information technology - Open Systems Interconnection - Systems Management: Security alarm reporting function".

3 Definitions and abbreviations

3.1 Definitions

In addition to the terms and definitions defined in 3GPP TS 32.111-1 [9], the following definitions apply to this document:

Event: It is an occurrence that is of significance to network operators, the NEs under surveillance and Network Management applications. Events do not have state.

IRPManager: defined in 3GPP TS 32.102 [7].

IRP document version number string: The IRP document version number (sometimes called "**IRPVersion**") string is used to identify a particular IRP solution set specification. It is derived using the following rule. Take the 3GPP document version number on the front page of the solution set specification, such as "3GPP TS 32.106-3 V3.2.0 (2000-12)". Discard the leading "3GPP TS ". Discard all characters after and including the last period. Eliminate leading and trailing spaces. Reduce multiple consecutive spaces with one space. Express the resultant in a string. Capitalised the string. For example, if the 3GPP document version number is "3GPP TS 32.106-3 V3.2.0 (2000-12)", then the IRP document version number shall be "32.106 V3.2".

Matching-Criteria-Attributes: It identifies a set of ITU-T Recommendation X.733 [2] defined attributes. Notifications carrying identical values for these attributes are considered to be carrying alarm information related to (a) the same network resource and (b) the same alarmed condition. The matching-criteria-attributes are: `objectInstance, eventType, probableCause and specificProblem`, if present.

Notification: It refers to the transport of events from `IRPAgent` to `IRPManager`. In this IRP, notifications are used to carry alarm information from `IRPAgent` to `IRPManager`.

`IRPAgent`: defined in 3GPP TS 32.102 [7].

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

CCITT	The International Telegraph and Telephone Consultative Committee
CMIP	Common Management Information Protocol
DN	Distinguished Name
EM	Element Manager
IOC	Information Object Class
IRP	Integration Reference Point
ITU-T	International Telecommunication Union, Telecommunication Sector
MO	Managed Object
MOC	Managed Object Class
MOI	Managed Object Instance
NE	Network Element
NM	Network Manager
OS	Operations System
OSI	Open System Interconnection
RDN	Relative Distinguished Name
SS	Solution Set
UML	Unified Modelling Language

4 Basic aspects

4.1 Background

Integration Reference Points (IRPs) are the means within 3G Telecom Management (TM) for specifying interoperable points of information exchange between systems and applications.

3GPP TS 32.101 [6] and 32.102 [7] contain background and introductory information about the IRP concept.

4.2 System Overview

The following figures identify system contexts of this document in terms of implementations called `IRPAgent` and `IRPManager`.

"`IRPManager`" depicts a process that interacts with `IRPAgent` for the purpose of receiving alarms via this IRP. Examples of `IRPManager` can be Network Management Systems and Alarm viewing devices (such as a local craft terminal). `IRPAgent` implements and supports the Alarm IRP.
`IRPAgent` can be one Network Element (NE) (see figure 2) or it can be one Element Manager (EM) with one or more NEs (see figure 1). In the latter case, the interfaces (represented by a thick dotted line) between the EM and the NEs are not subject of this IRP. Whether EM and NE share the same hardware system is not relevant to this document either. By observing the interaction across the Alarm IRP, one cannot deduce if EM and NE are integrated in a single system or if they run in separate systems.

As indicated in figure 1 and figure 2, the subject document need to be complemented with the Notification IRP [5] (to allow `IRPManager` to subscribe to notifications issued by `IRPAgent` and (optionally) product-specific resource models describing the MOs maintained by the `IRPAgent`).

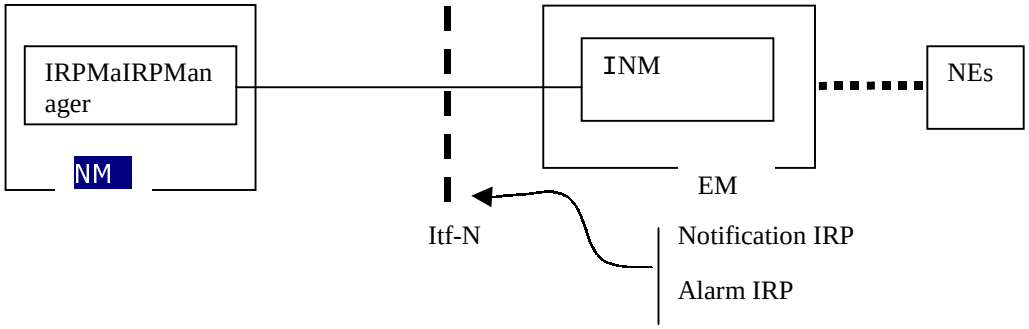


Figure 1: System Context A

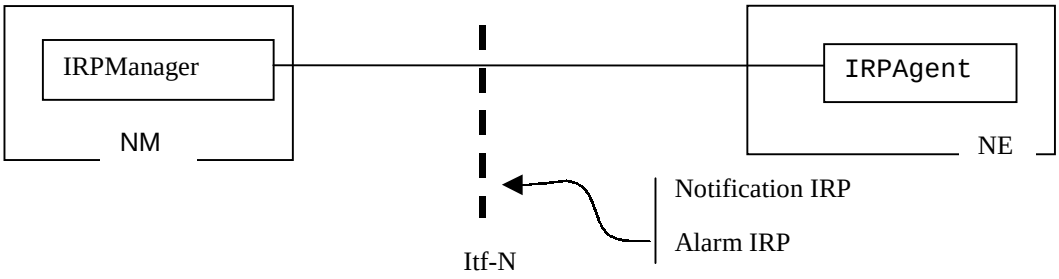


Figure 2: System Context B

5 Information Object Classes

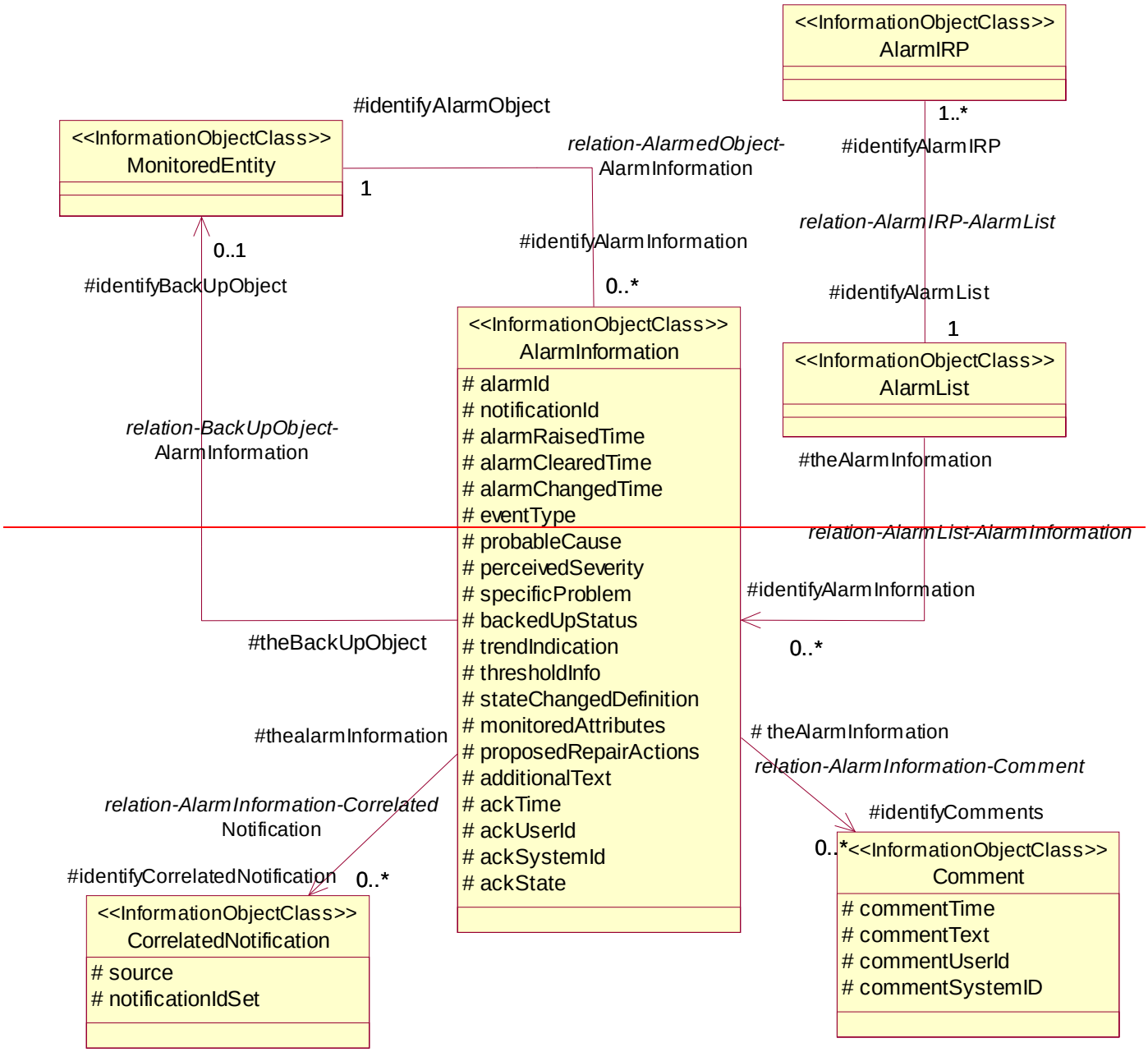
5.1 Information entities imported and local label

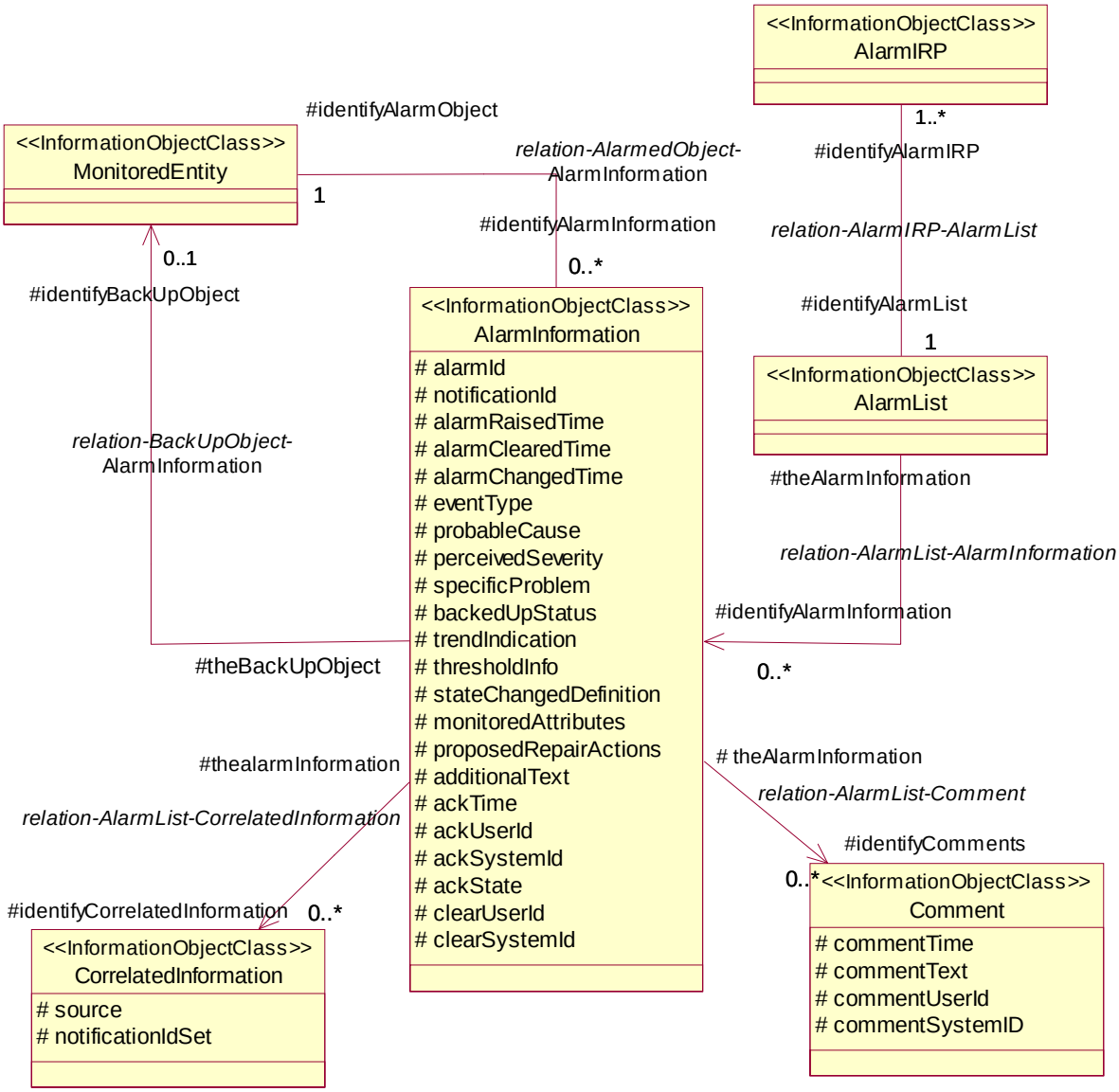
Label reference	Local label
32.302 [5], information object class, NotificationIRP	NotificationIRP
32.302 [5], interface, notificationIRPNotification	notificationIRPNotification
32.622 [10], information object class, IRPAgent	IRPAgent
32.622 [10], information object class, ManagedGenericIRP	ManagedGenericIRP

5.2 Class diagram

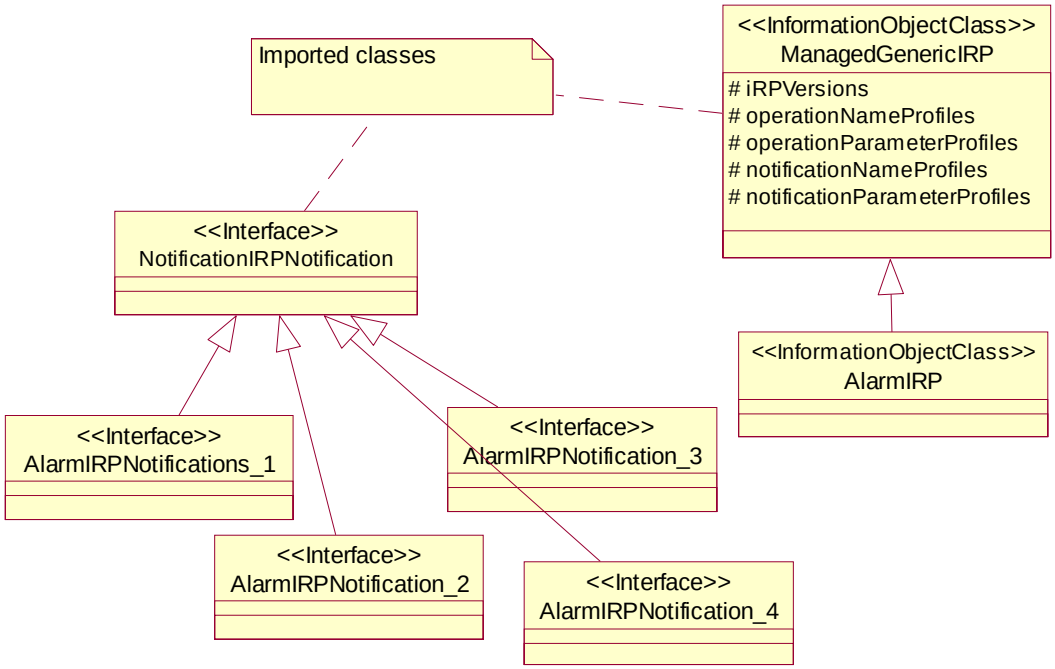
This sub-clause introduces the set of information object classes (IOCs) that encapsulate information within the `IRPAgent`. The intent is to identify the information required for the AlarmIRP Agent implementation of its operations and notification emission. This sub-clause provides the overview of all support object classes in UML. Subsequent sub-clauses provide more detailed specification of various aspects of these support object classes.

5.2.1 Attributes and relationships




<<InformationObjectClass>>
AlarmIRP
1..*
#identifyAlarmIRP
relation-AlarmIRP-AlarmList
#identifyAlarmList
1
<<InformationObjectClass>>
AlarmList
#theAlarmInformation
relation-AlarmList-AlarmInformation
#identifyAlarmInformation
0..*
#identifyAlarmObject
<<InformationObjectClass>>
MonitoredEntity
1
relation-AlarmedObject-
AlarmInformation
#identifyAlarmInformation
0..*
0..1
#identifyBackUpObject
relation-BackUpObject-
AlarmInformation
#theBackUpObject
<<InformationObjectClass>>
AlarmInformation
alarmId
notificationId
alarmRaisedTime
alarmClearedTime
alarmChangedTime
eventType
probableCause
perceivedSeverity
specificProblem
backedUpStatus
trendIndication
thresholdInfo
stateChangedDefinition
monitoredAttributes
proposedRepairActions
additionalText
ackTime
ackUserId
ackSystemId
ackState
clearUserId
clearSystemId
#thealarmInformation
relation-AlarmList-CorrelatedInformation
#identifyCorrelatedInformation
0..*
<<InformationObjectClass>>
CorrelatedInformation
source
notificationIdSet
theAlarmInformation
relation-AlarmList-Comment
#identifyComments
0..*
<<InformationObjectClass>>
Comment
commentTime
commentText
commentUserId
commentSystemID

5.2.2 Inheritance



5.3 Information Object Class Definitions

5.3.13 AlarmInformation

5.3.1.1 Definition

AlarmInformation contains information about alarm condition of an alarmed MonitoredEntity.

One IRPAgent is related to at most one AlarmList. The IRPAgent or its related AlarmIRP or the related AlarmList assigns an identifier, called alarmId, to each AlarmInformation in the AlarmList. An alarmId unambiguously identifies one AlarmInformation in the AlarmList.

5.3.1.2 Attribute

Attribute name	Support Qualifier
alarmId	M
notificationId (note 1)	M
alarmRaisedTime	M
alarmClearedTime	M
alarmChangedTime	O
eventType	M
probableCause	M
perceivedSeverity	M
specificProblem	O
backedUpStatus	O
trendIndication	O
thresholdInfo	O
stateChangedDefinition	O
monitoredAttributes	O
proposedRepairActions	O
additionalText	O
ackTime	M
ackUserId	M
ackSystemId	O
ackState	M
clearUserId	M (note 2)
clearSystemId	O (note 2)

Note 1: This attribute may be "retired/removed" in Release 5 when Log IRP is introduced. Its removal implies that information carried in this attribute is no longer made accessible to IRPManager via the getAlarmList().

Note 2: These attributes and qualifiers are applicable only if the IRPAgent supports clearAlarms() (they are absent if clearAlarms() is not supported).

5.3.1.3 State diagram

Alarms have states. The alarm state information is captured in AlarmInformation in AlarmList.

The solid circle icon represents the Start State. The double circle icon represents the End State. In this state, the alarm is Cleared and acknowledged. The AlarmInformation shall not be accessible via the IRP and is removed from the AlarmList.

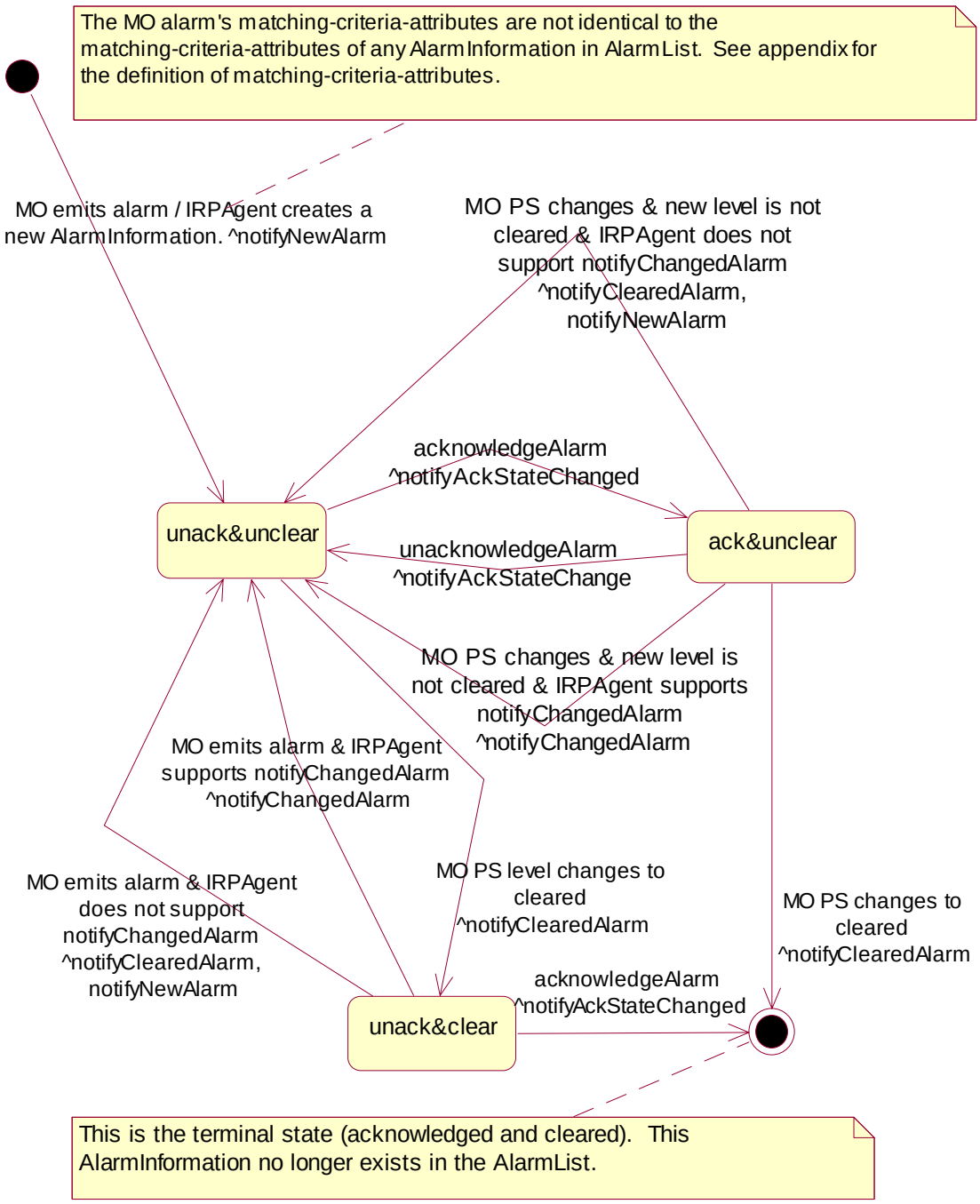
Note the state diagram uses " X / Y ^ Z " to label the arc that indicates state transition. The meanings of X, Y and Z are:

- X identifies the triggering event
- Y identifies the action of IRPAgent because of the triggering event
- Z is the notification to be emitted by IRPAgent because of the triggering event

Note that `acknowledgeAlarm^notifyAckStateChanged` and the `unacknowledgeAlarm^notifyAckStateChange` refer to cases when the request of the `IRPManager` is successful for the `AlarmInformation` concerned. They do not refer to the cases when the request is a failure since in the failure cases, no state transition would occur.

Note that, to reduce cluttering to the diagram, the `setComment^notifyComment` is not included in the figure. One transition should be applied from `unack&unclear` to itself. Similarly, another transition should be applied from `ack&unclear` to itself. Another one is from `unack&clear` to itself.

Note that "PS" used in the state diagram stands for "perceived severity".



5.3.2 AlarmList

5.3.2.1 Definition

IRPAgent maintains an AlarmList. It contains all currently active alarms (i.e. AlarmInformation whose perceivedSeverity is not Cleared) and alarms that are Cleared but not yet acknowledged.

5.3.2.2 Attribute

There is no additional attribute defined for this IOC besides those inherited.

5.3.3 AlarmIRP

5.3.3.1 Definition

AlarmIRP is the representation of the alarm management capabilities specified by this specification. This IOC inherits from ManagedGenericIRP IOC specified in [14].

5.3.4 Comment

5.3.4.1 Definition

Comment contains commentary and associated information such as the time when the commentary is made.

5.3.4.2 Attribute

Attribute Name	Support Qualifier
commentTime	M
commentText	M
commentUserId	M
commentSystemId	O

5.3.5 CorrelatedNotification

5.3.5.1 Definition

It identifies one MonitoredEntity. For that MonitoredEntity identified, a set of notification identifiers is also identified. One or more CorrelatedNotification instances can be related to an AlarmInformation. In this case, the information of the AlarmInformation is said to be correlated to information carried in the notifications identified by the CorrelatedNotification instances. See further definition of correlated notification in ITU-T Recommendation X.733 [2] clause 8.1.2.9.

The meaning of correlation is dependent on the type of notification itself. See the comment column of the correlatedNotification input parameter for each type of notification, such as notifyNewAlarm.

Notification carries AlarmInformation. The AlarmInformation instances referred to by the correlatedNotification may or may not exist in the AlarmList. For example, the AlarmInformation carried by the identified notification may have been acknowledged and Cleared and therefore, no longer exist in the AlarmList.

5.3.5.2 Attribute

Attribute Name	Support Qualifier
source	M
notificationIdSet	M

5.3.6 MonitoredEntity

5.3.6.1 Definition

It encapsulates a subset of information of an IOC that can emit alarms. It can also encapsulate a subset of information of an IOC that serves as the back up object.

5.3.6.2 Attribute

There is no attribute for this IOC.

5.4 Information relationships definition

5.4.1 relation-AlarmIRP-AlarmList (M)

5.4.1.1 Definition

This represents the relationship between AlarmIRP and AlarmList.

5.4.1.2 Role

Name	Definition
identifyAlarmIRP	It represents the capability to obtain the identities of one or more AlarmIRP.
identifyAlarmList	It represents the capability to obtain the identity of one AlarmList.

5.4.1.3 Constraint

There is no constraint for this relationship.

5.4.2 relation-AlarmList-AlarmInformation (M)

5.4.2.1 Definition

This represents the relationship between AlarmList and AlarmInformation.

5.4.2.2 Role

Name	Definition
theAlarmInformation	It represents the AlarmInformation.
identifyAlarmInformation	It represents a capability to obtain the information contained in AlarmInformation.

5.4.2.3 Constraint

Name	Definition
inv_ hasAlarmInformation1	No AlarmInformation playing the role of theAlarmInformation shall have its perceivedSeverity = “cleared” and its ackState = “acknowledged”.
inv_ hasAlarmInformation2	The alarmId of all AlarmInformation instances playing the role of theAlarmInformation are distinct.

5.4.3 relation-AlarmInformation-Comment (M)

5.4.3.1 Definition

This represents the relationship between `AlarmInformation` and `Comment`.

5.4.3.2 Role

Name	Definition
theAlarmInformation	It represents the AlarmInformation.
identifyComment	It represents a capability to obtain the information contained in Comment.

5.4.3.3 Constraint

There is no constraint.

5.4.4 relation-AlarmInformation-CorrelatedNotification (M)

5.4.4.1 Definition

This represents the relationship between `AlarmInformation` and `CorrelatedNotification`.

5.4.4.2 Role

Name	Definition
theAlarmInformation	It represents the AlarmInformation.
identifyCorrelatedNotification	It represents a capability to obtain the information contained in CorrelatedNotification.

5.4.4.3 Constraint

There is no constraint.

5.4.5 relation-AlarmedObject-AlarmInformation (M)

5.4.5.1 Definition

This represents the relationship between `MonitoredEntity` and `AlarmInformation`.

5.4.5.2 Role

Name	Definition
identifyAlarmedObject	It represents the capability to obtain the identification, in terms of objectClass and objectInstance, of alarmed network resource.
identifyAlarmInformation	It represents the capability to obtain the identities of AlarmInformation.

5.4.5.3 Constraint

Name	Definition
inv_relation-AI-ME	All AlarmInformation involved in this relationship with the same MonitoredEntity shall have at least one different value in the following attributes: eventType, probableCause and specificProblem.

5.4.6 `relation-backUpObject-AlarmInformation (O)`

5.4.6.1 Definition

The relationship represents the relationship between `AlarmInformation` and the `backUpObject`.

5.4.6.2 Role

Name	Definition
identifyBackUpObject	It represents a capability to obtain the identification, in terms of objectClass and objectInstance, of the backUpObject.

5.4.6.3 Constraint

Name	Definition
inv_identifyBackUpObject	This relationship is present if and only if the AlarmInformation.backedUpStatus attribute is present and is indicating true.

5.5 Information attribute definition

5.5.1 Definition and legal values

Name	Definition	Legal Values
alarmId	It identifies one AlarmInformation in the AlarmList.	
notification Id	It identifies the notification that carries the AlarmInformation.	
alarmRaised Time	It indicates the date and time when the alarm is first raised by the alarmed resource.	All values indicating valid time.
alarmChanged Time	It indicates the last date and time when the AlarmInformation is changed by the alarmed resource. Changes to AlarmInformation caused by invocations of the IRPManager would not change this date and time.	All values indicating valid time.
alarmCleared Time	It indicates the date and time when the alarm is Cleared.	All values indicating valid time.
eventType	It indicates the type of event. See Annex A for information on event type.	See Annex A.
probableCause	It qualifies alarm and provides further information than eventType. See Annex B for a complete listing.	See Annex B.
perceived Severity	It indicates the relative level of urgency for operator attention.	Critical, Major, Minor, Warning, Indeterminate, Cleared: see ITU-T Recommendation X.733 [2]. This IRP does not recommend the use of indeterminate.
specific Problem	It provides further qualification on the alarm than probableCause. This attribute value shall be single-value and of simple type such as integer or string. See definition in ITU-T Recommendation X.733 [2] clause 8.1.2.2.	Provided by vendor.
backedUp Status	It indicates if an object (the MonitoredEntity) has a back up. See definition in ITU-T Recommendation X.733 [2] clause 8.1.2.4.	All values that carry the semantics of backedUpStatus defined by ITU-T X.733 [2] clause 8.1.2.4.
trend Indication	It indicates if some observed condition is getting better, worse, or not changing.	“Less severe”, “no change”, “more severe”: see definition in ITU-T Recommendation X.733 [2] clause 8.1.2.6.
thresholdInfo	It indicates the direction of threshold crossing.	See definitions in ITU-T Recommendation X.733 [2] clause 8.1.2.7.
stateChange Definition	It indicates MO attribute value changes. See definition in ITU-T Recommendation X.733 [2] clause 8.1.2.10.	
monitored Attributes	It indicates MO attributes whose value changes are being monitored. See definition in ITU-T Recommendation X.733 [2] clause 8.1.2.11.	
proposed RepairActions	It indicates proposed repair actions. See definition in ITU-T Recommendation X.733 [2] clause 8.1.2.12.	
additional Text	It carries semantics that is outside the scope of this IRP specification. It may provide the identity of the NE (e.g. RNC, Node-B) from which the alarm has been originated. It corresponds to the “user label” attribute of the object class representing the NE in the Generic Network Resource Model [10]. It can contain further information on the alarm.	N/A
ackTime	It identifies the time of last operation acknowledgeAlarms or unacknowledgeAlarms.	All values that indicate valid time that are later than that carried in alarmRaisedTime.

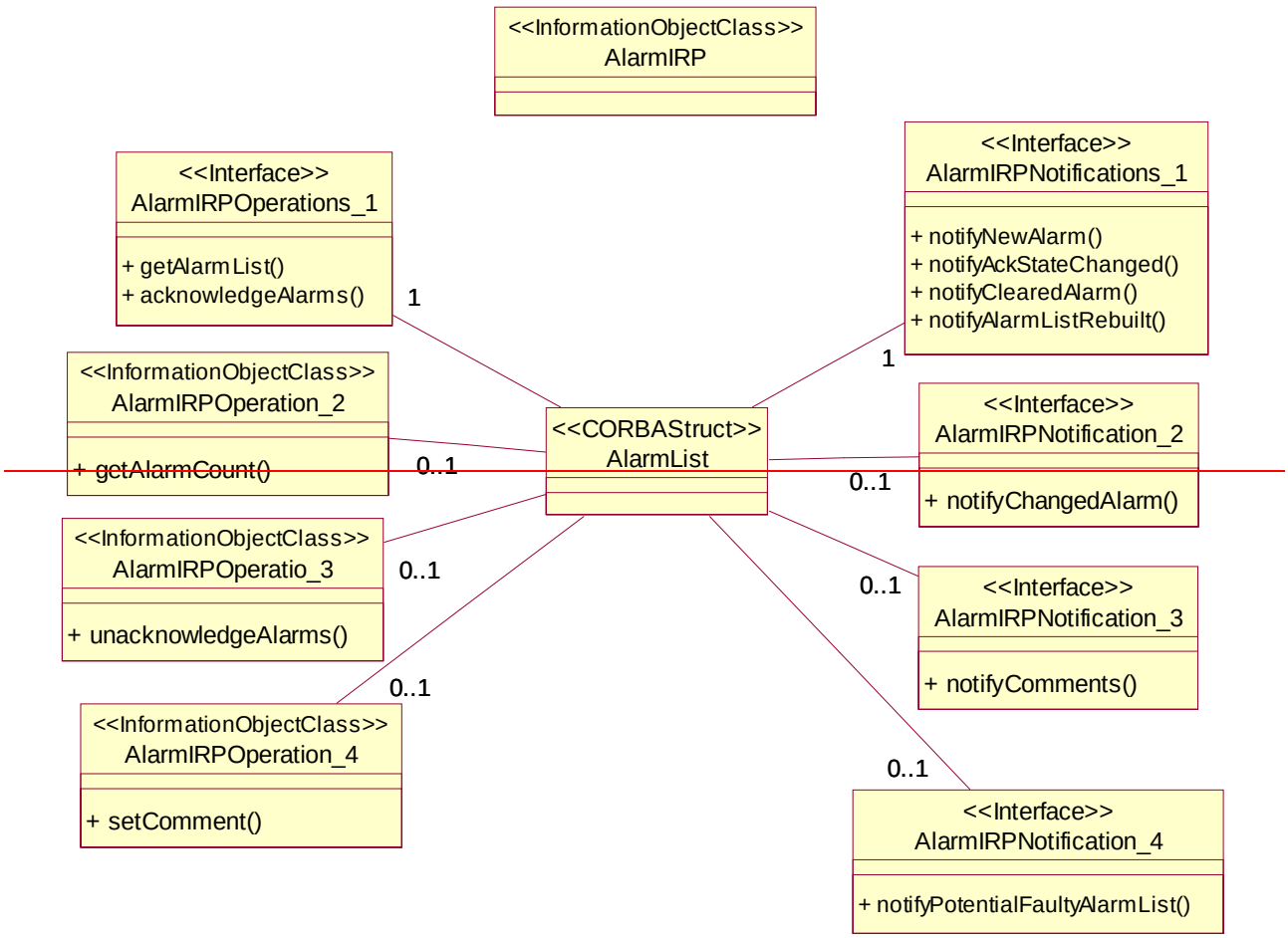
Name	Definition	Legal Values
ackUserId	It identifies the last user who has change the Acknowledgement State via operation acknowledgeAlarms or unacknowledgeAlarms. ~~It can be used to identify the human operator such as "John Smith" or it can identify a group, such as "Team Six", or it can contain no information such as "".~~	~~N/A~~It can be used to identify the human operator such as "John Smith" or it can identify a group, such as "Team Six", or it can contain no information such as "".
ackSystemId	It identifies the system in which IRPManager, that invokes the acknowledgeAlarms or unacknowledgeAlarms operation, runs.	~~N/A~~ It can be used to identify the system, such as "system 6" or it can contain no information such as "".
ackState	It identifies the Acknowledgement State of the alarm.	Acknowledged: the alarm has been acknowledged. Unacknowledged: the alarm has been unacknowledged or the alarm has never been acknowledged.
commentTime	It carries the time when a comment is made via setComment operation.	
commentText	It carries the textual comment made via setComment operation.	
commentUserId	It carries the identification of the user who made the comment via setComment operation.	
commentSystemId	It carries the identification of the system in which the IRPManager runs. That IRPManager supports the user that made the comment.	
source	It identifies one MonitoredEntity.	All values that carry the semantics of DN.
notificationIdSet	It carries one or more notification identifiers.	
clearUserId	It carries the identity of the user who invokes the clearAlarms operation.	It can be used to identify the human operator such as "John Smith" or it can identify a group, such as "Team Six", or it can contain no information such as "".
clearSystemId	It carries the identity of the system in which the IRPManager runs. That IRPManager supports the user who invokes the clearAlarms().	It can be used to identify the system, such as "system 6" or it can contain no information such as "".

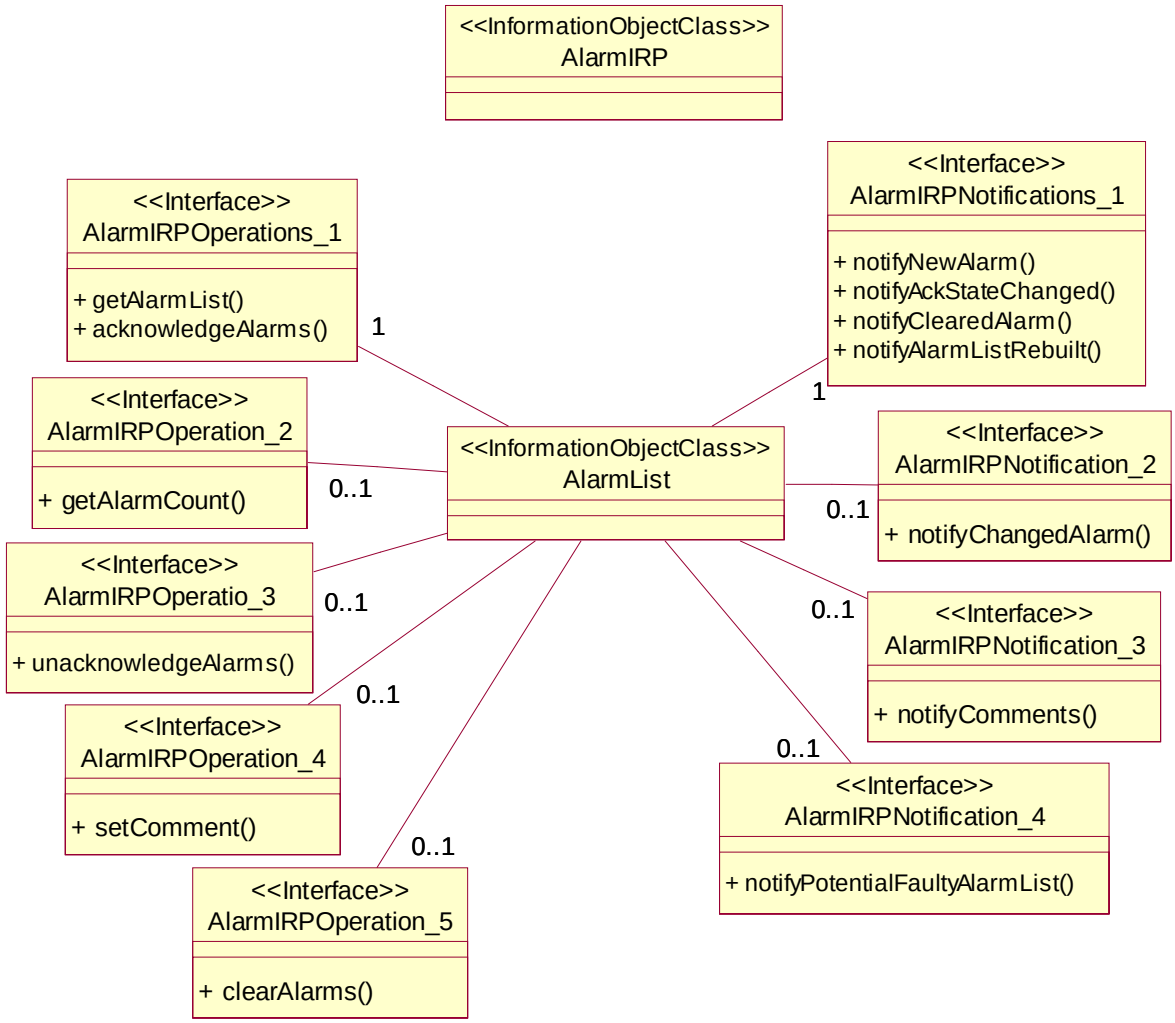
5.5.2 Constraints

Name	Definition
inv_alarmChangedTime	Time indicated shall be later than that carried in alarmRaisedTime.
inv_alarmClearedTime	Time indicated shall be later than that carried in alarmRaisedTime.
inv_ackTime	Time indicated shall be later than that carried in alarmRaisedTime.
inv_notificationId	NotificationIds shall be chosen to be unique across all notifications of a particular managed object (representing the NE) throughout the time that alarm correlation is significant. The algorithm by which alarm correlation is accomplished is outside the scope of this IRP.

6 Interface Definition

6.1 Class diagram





6.2 Generic rules

Rule 1: each operation with at least one input parameter supports a pre-condition valid_input_parameter which indicates that all input parameters shall be valid with regards to their information type. Additionally, each such operation supports an exception operation_failed_invalid_input_parameter which is raised when pre-condition valid_input_parameter is false. The exception has the same entry and exit state.

Rule 2: Each operation with at least one optional input parameter supports a set of pre-conditions supported_optional_input_parameter_xxx where "xxx" is the name of the optional input parameter and the pre-condition indicates that the operation supports the named optional input parameter. Additionally, each such operation supports an exception operation_failed_unsupported_optional_input_parameter_xxx which is raised when (a) the pre-condition supported_optional_input_parameter_xxx is false and (b) the named optional input parameter is carrying information. The exception has the same entry and exit state.

Rule 3: each operation shall support a generic exception operation_failed_internal_problem that is raised when an internal problem occurs and that the operation cannot be completed. The exception has the same entry and exit state.

6.3 Interface AlarmIRPOperations_1

6.3.1 acknowledgeAlarms (M)

6.3.1.1 Definition

The IRPManager invokes this operation to acknowledge one or more alarms.

6.3.1.2 Input Parameters

Name	Qualifier	Information Type	Comment
alarmInformationAndSeverityReferenceList	M	List of AlarmInformation.alarmId and AlarmInformation.perceivedSeverity	It carries one or more identifiers identifying AlarmInformation instances in AlarmList, including optionally the perceivedSeverity of the AlarmInformation instance that is going to be acknowledged. alarm InformationAndSeverity ReferenceList {alarmId - Mandatory; perceivedSeverity - Optional }
AckUserId	M	AlarmInformation.ackUserId	It identities the user acknowledging the alarm.
ackSystemId	O	AlarmInformation.ackSystemId	It identifies the processing system on which the subject IRPManager runs. It may be absent ~~contain no information~~ implying that IRPManager does not wish this information be kept in AlarmInformation in AlarmList.

6.3.1.3 Output Parameters

Name	Qualifier	Matching Information	Comment
badAlarm Information ReferenceList	M	List of pair of AlarmInformation.alarmId, ENUM (UnknownAlarmId, AcknowledgmentFailed, WrongPerceivedSeverity) and additional failure reason.	If allAlarmsAcknowledged is true, it contains no information. If someAlarmAcknowledged is true, then it contains identifications of AlarmInformation that are (a) present in input parameter AlarmInformationReferenceList but are absent in the AlarmList = UnknownAlarmId; or (b) present in input parameter AlarmInformationReferenceList and are present in the AlarmList but the Acknowledgement Information (see note below table) has not changed, in contrast to IRPManager's request = AcknowledgmentFailed; or (c) present in input parameter AlarmInformationReferenceList and are present in the AlarmList but the perceivedSeverity to be acknowledged has changed and/or is different within the Alarm List = WrongPerceivedSeverity (applicable only if perceivedSeverity was provided).
status	M	ENUM (OperationSucceeded, OperationFailed, OperationPartiallySucceeded)	If someAlarmAcknowledged is true, status = OperationPartiallySuceeded. If allAlarmsAcknowledged is true, status = OperationSucceeded. If operation_failed is true, status = OperationFailed.

NOTE: Acknowledgement Information is defined as the information contained in `AlarmInformation.`ackTime, `AlarmInformation.`ackUserId, AlarmInformaton.ackSystemId, `AlarmInformation.`ackState.

6.3.1.4 Pre-condition

`atLeastOneValidId.`

Assertion Name	Definition
atLeastOneValidId	The AlarmInformationReferenceList contains at least one identifier that identifies one AlarmInformation in AlarmList and that this identified AlarmInformation shall have its ackState indicating “unacknowledged” and, if provided, an equal perceivedSeverity.

6.3.1.5 Post-condition

`someAlarmAcknowledged OR allAlarmsAcknowledged.`

Assertion Name	Definition
someAlarmAcknowledged	At least one but not all AlarmInformation identified in input parameter AlarmInformationReferenceList has been acknowledged. Acknowledgement of an AlarmInformation means that the ackState attribute has been set to “acknowledged”, that ackUserId, ackSystemId attributes of this AlarmInformation have been set to the values provided as input parameter and that the time of acknowledgeAlarms operation has been registered in ackTime attribute.
allAlarmsAcknowledged	All AlarmInformation identified in input parameter have been acknowledged. Acknowledgement of an AlarmInformation means that the ackState attribute has been set to “acknowledged”, that ackUserId, ackSystemId attributes of this AlarmInformation have been set to the values provided as input parameter and that the time of acknowledgeAlarms operation has been registered in ackTime attribute.

6.3.1.6 Exceptions

Name	Definition
operation_failed	**Condition:** Pre-condition is false or post-condition is false. **Returned Information:** The output parameter status. **Exit state:** Entry state.

6.3.2 `getAlarmList` (M)

6.3.2.1 Definition

`IRPManager` requests IRPAgent to provide the list of `AlarmInformation` instances in `AlarmList`.

There are two modes of operation. One mode is synchronous. In this mode, the list of `AlarmInformation instances` in `AlarmList` is returned synchronously with the operation. The other mode is asynchronous. In this mode, the list of `AlarmInformation` instances is returned via notifications. In asynchronous mode of operation, the only information returned synchronously is the status of the operation. The mode of operation to be used is determined by means outside the scope of specification. To use asynchronous mode, the `IRPManager` must have established a subscription with the IRPAgent notificationIRP via the `subscribe` operation specified in [5].

6.3.2.2 Input Parameters

Name	Qualifier	Information Type	Comment
alarmAckState	O	ENUM (all alarms, all active alarms, all active and acknowledged alarms, all active and unacknowledged, all Cleared and unacknowledged alarms, all unacknowledged)	It carries a constraint. The IRPAgent shall apply it on AlarmInformation instances in AlarmList when constructing its output parameter AlarmInformationList.
filter	O	N/A	It carries a filter constraint. The IRPAgent shall apply it on AlarmInformation instances in AlarmList when constructing its output parameter AlarmInformationList.

6.3.2.3 Output Parameters

Name	Qualifier	Matching Information	Comment
AlarmInformationList	M	List of AlarmInformation.	It carries AlarmInformation in AlarmList. Case when synchronous mode of operation is used: (a) The IRPAgent shall apply the constraints expressed in alarmAckState and filter to AlarmInformation instances when constructing this output parameter. Case when asynchronous mode of operation is used (i.e., this output parameter is conveyed via notifications): (a) If the filter parameter is present, the IRPAgent shall apply the constraint when constructing this output parameter. Furthermore, if the alarmAckState constraint is present, the IRPAgent shall apply that constraint as well. The filter constraint, if any, that is currently active in the notification channel is not used for the construction of this output parameter. (b) If the filter parameter is absent, the IRPAgent shall apply the filter constraint currently active in the notification channel when constructing this output parameter. If the alarmAckState constraint is present, the IRPAgent shall apply that constraint as well.
status	M	ENUM (OperationSucceeded, OperationFailed)	If allAlarmInformationReturned is true, status = OperationSucceeded. If operation_failed is true, status = OperationFailed.

6.3.2.4 Pre-condition

There is no pre-condition.

6.3.2.5 Post-condition

`allAlarmInformationReturned.`

Assertion Name	Definition
allAlarmInformationReturned	All AlarmInformation that satisfy the constraints expressed in input parameters filter and alarmAckState and are present in the AlarmList at the moment of this operation invocation are returned. All AlarmInformation in AlarmList remains unchanged as the result of this operation.

6.3.2.6 Exceptions

Assertion Name	Definition
operation_failed	**Condition:** At least one input parameter is invalid or the pre-condition is false or the post-condition is not true. **Returned Information:** The output parameter status. **Exit state:** Entry state.

6.4 Interface `AlarmIRPOperation_2`

6.4.1 `getAlarmCount` (O)

6.4.1.1 Definition

An `IRPManager` wishes to know the amount of `AlarmInformation` kept in the AlarmList. The `IRPManager` requests the counts via this operation. Possible usage is for `IRPManager` to find out the number of `AlarmInformation` in `AlarmList` before invoking `getAlarmList` operation.

6.4.1.2 Input Parameters

Name	Qualifier	Information Type	Comment
filter	O	N/A	It carries a filter constraint. The operation shall apply it when counting the AlarmInformation instances in AlarmList. Case when synchronous mode of operation is used for getAlarmList: (a) If this parameter is present, the operation shall count the AlarmInformation instances which satisfy both (a) this filter constraint and (b) the condition set by input parameter alarmAckState. (b) If this parameter is absent, the operation shall count all AlarmInformation instances that satisfy the condition set by input parameter alarmAckState. Case when asynchronous mode of operation is used for getAlarmList: (a) If this parameter is present, the operation shall count all AlarmInformation instances that satisfy this filter constraint and the condition set by input parameter alarmAckState. (b) If this parameter is absent, the operation shall count AlarmInformation instances that satisfy (a) the filter constraint currently active in the notification channel established between the IRPManager and the IRPAgent that is equipped with NotificationIRP capabilities and (b) the condition set by input parameter alarmAckState.
alarmAckState	O	ENUM (all alarms, all active alarms, all active and acknowledged alarms, all active and unacknowledged, all cleared and unacknowledged alarms, all unacknowledged)	It carries a constraint. The operation shall apply it on AlarmInformation instances in AlarmList when counting.

6.4.1.3 Output Parameters

Name	Qualifier	Matching Information	Comment
criticalCount, majorCount, minorCount, warningCount, indeterminateCount, clearedCount	M	N/A	They carry the number of AlarmInformation in AlarmList that has the following properties. Case when synchronous mode of operation is used: (a) The operation shall apply the constraints expressed in alarmAckState and filter to AlarmInformation instances when counting. Case when asynchronous mode of operation is used (i.e., this output parameter is conveyed via notifications): (a) If the filter parameter is present, the operation shall apply the constraint when counting. Furthermore, if the alarmAckState constraint is present, the operation shall apply that constraint as well. The filter constraint, if any, that is currently active in the notification channel is not used for the counting. (b) If the filter parameter is absent, the operation shall apply the filter constraint currently active in the notification channel when counting. If the alarmAckState constraint is present, the operation shall apply that constraint as well.
status	M	ENUM (OperationSucceeded, OperationFailed)	If allAlarmInformationCounted is true, status = OperationSucceeded. If operation_failed is true, status = OperationFailed.

6.4.1.4 Pre-condition

There are no pre-conditions.

6.4.1.5 Post-condition

`allAlarmInformationCounted.`

Assertion Name	Definition
allAlarmInformationCounted	All AlarmInformation that satisfy the constraints expressed in input parameters filter and alarmAckState and are present in the AlarmList at the moment of this operation invocation are counted and the result returned. All AlarmInformation in AlarmList remains unchanged as the result of this operation.

6.4.1.6 Exceptions

Name	Definition
operation_failed	**Condition:** the pre-condition is false or the post-condition is true. **Returned Information:** The output parameter status. **Exit state:** Entry state.

6.5 Interface AlarmIRPOperation_3

6.5.1 unacknowledgeAlarms (O)

6.5.1.1 Definition

IRPManager invokes this operation to remove acknowledgement information kept in one or more AlarmInformation instances.

6.5.1.2 Input Parameters

Name	Qualifier	Information Type	Comment
alarm InformationReferenceList	M	List of AlarmInformation.alarmId	It carries one or more identifiers identifying AlarmInformation in AlarmList.
ackUserId	M	AlarmInformation.ackUserId	It identities the user that invokes this operation.
ackSystemId	O	AlarmInformation.ackSystemId	It identifies the processing system on which the subject IRPManager runs.

6.5.1.3 Output Parameters

Name	Qualifier	Matching Information	Comment
badAlarmInformationReferenceList	M	List of pair of AlarmInformation.alarmId and the failure reason.	If allAlarmsUnacknowledged is true, it contains no information. If someAlarmUnacknowledged is true, then it contains identifications of AlarmInformation that are (a) present in input parameter AlarmInformationReferenceList but are absent in the AlarmList; or (b) present in input parameter AlarmInformationReferenceList and are present in the AlarmList but the Acknowledgement Information (see note below table) has not changed, in contrast to IRPManager's request.
status	M	ENUM (OperationSucceeded, OperationFailed, OperationPartiallySucceeded)	If someAlarmUnacknowledged is true, status = OperationPartiallySuceeded. If allAlarmsUnacknowledged is true, status = OperationSucceeded. If operation_failed is true, status = OperationFailed.

NOTE: Acknowledgement Information is defined as the information contained in AlarmInformation.ackTime, AlarmInformation.ackUserId, AlarmInformaton.ackSystemId and AlarmInformation.ackState.

6.5.1.4 Pre-condition

atLeastOneValidId AND validUserId&SystemId.

Assertion Name	Definition
atLeastOneValidId	The AlarmInformationReferenceList contains at least one identifier that identifies one AlarmInformation in AlarmList and that this identified AlarmInformation shall have its ackState indicating "acknowledged".
validUserId&SystemId	The values of ackUserId and ackSystemId attributes of the AlarmInformation must be the same as the ones provided as input parameters. The AlarmInformation is identified by the input parameter AlarmInformationReferenceList.

6.5.1.5 Post-condition

`someAlarmUnacknowledged` OR `allAlarmsUnacknowledged`.

Assertion Name	Definition
someAlarmUnacknowledged	At least one but not all AlarmInformation identified in input parameter alarmListReferenceList has been unacknowledged. This means that the ackState attribute has been set to "unacknowledged", that ackTime, ackUserId, ackSystemId attributes of this AlarmInformation have been set to containing no information.
allAlarmsUnacknowledged	All AlarmInformation identified in input parameter have been unacknowledged. This means that the ackState attribute has been set to "unacknowledged", that ackTime, ackUserId, ackSystemId attributes of this AlarmInformation have been set to contain no information.

6.5.1.6 Exceptions

Name	Definition
operation_failed	**Condition:** Pre-condition is false or post-condition is false. **Returned Information:** The output parameter status. **Exit state:** Entry state.

6.6 Interface `AlarmIRPOperation_4`

6.6.1 `setComment` (O)

6.6.1.1 Definition

The `IRPManager` invokes this operation to record a comment in one or more `AlarmInformation` instances in AlarmList.

6.6.1.2 Input Parameters

Name	Qualifier	Information Type	Comment
AlarmInformation ReferenceList	M	List of AlarmInformation.alarmId	It carries one or more identifiers identifying AlarmInformation instances in the AlarmList.
commentUserId	M	The Comment.commentUserId where Comment is involved in relation-AlarmInformation-Comment with an AlarmInformation.	
commentSystemId	O	The Comment.commentSystemId where Comment is involved in relation-AlarmInformation-Comment with an AlarmInformation.	
commentText	M	The comment.commentText where Comment is involved in relation-AlarmInformation-Comment with an AlarmInformation.	

6.6.1.3 Output Parameter

Name	Qualifier	Matching Information	Comment
badAlarm Information ReferenceList	M	List of pair of AlarmInformation.alarmId and the failure reason.	If allUpdated is true, it contains no information. If someUpdated is true, then it contains identifications of AlarmInformation that are not present in AlarmList or that they are present, but AlarmInformation.comments has not changed, in contrast to IRPManager's request.
Status	M	ENUM(Operation succeeded, Operation failed, Operation partially failed)	If allUpdated is true, then status = OperationSsucceeded. If someUpdated is true, then status = OperationPartiallyFailed. If exception operationFailed is raised, then status = OperationFailed.

6.6.1.4 Pre-condition

`atLeastOneValidId.`

Assertion Name	Properties
atLeastOneValidId	The AlarmInformationReferenceList contains at least one identifier that identifies one AlarmInformation in AlarmList.

6.6.1.5 Post-condition

`allUpdated OR someUpdated`.

Assertion Name	Properties
allUpdated	The AlarmInformation.comment of all alarms identified by the input parameter AlarmInformationReferenceList has been updated. The input parameter commentText, commentUserId and commentSystemId are added to the AlarmInformation.comment. The time of the operation invocation is captured in the AlarmInformation.comment as well. To make it possible to add the new comment, the IRPAgent may remove one or more old comment previously held by AlarmInformation.comments.
someUpdated	The AlarmInformation.comment attribute of at least one but not all alarms identified by the input parameter AlarmInformationReferenceList has been updated. The input parameter commentText, commentUserId and commentSystemId are added to the AlarmInformation.comment. The time of the operation invocation is captured in the AlarmInformation.comment as well. To add a new Comment, it may be necessary to remove one or more old Comment instances being held. The commentTime of the removed Comment instances shall be older than that of the remaining Comment instances.

6.6.1.6 Exceptions

Name	Properties
operation_failed	**Condition:** the pre-condition is false or the post-condition is false. **Returned Information:** The output parameter status. **Exit state:** Entry state.

6.7 Interface AlarmIRPOperation_5

6.7.1 clearAlarms (O)

6.7.1.1 Definition

The IRPManager invokes this operation to clear one or more AlarmInformation instances in AlarmList. For example, this operation can be used to support the manual clearing of the ADMC (automatic detection and manual clearing, see also [9]) alarms.

6.7.1.2 Input Parameter

Name	Qualifier	Information Type	Comment
alarmInformationReferenceList	M	List of AlarmInformation.alarmId	It carries one or more identifiers identifying AlarmInformation instances in the AlarmList.
clearUserId	M	N/A	It identities the user clearing the alarm.
clearSystemId	O	N/A	It identifies the processing system on which the subject IRPManager runs. It may be absent implying that IRPManager does not wish this information be known to the IRPAgent.

6.7.1.3 Output Parameter

Name	Qualifier	Matching Information	Comment
badAlarmInformationReferenceList	M	List of pair of AlarmInformation.alarmId and the failure reason.	If allCleared is true, it contains no information. If someCleared is true, then it contains identifications of AlarmInformation that are not present in AlarmList or that are present in AlarmList but remain unchanged, in contrast to IRPManager's request.
Status	M	ENUM(Operation succeeded, Operation failed, Operation partially failed)	If allCleared is true, then status = OperationSucceeded. If someCleared is true, then status = OperationPartiallyFailed. If exception operationFailed is raised, then status = OperationFailed.

6.7.1.4 Pre-condition

atLeastOneValidId.

Assertion Name	Properties
atLeastOneValidId	The input parameter alarmInformationReferenceList contains at least one identifier that identifies one AlarmInformation in AlarmList.

6.7.1.5 Post-condition

`allCleared OR someCleared.`

Assertion Name	Properties
allCleared	The AlarmInformation.perceivedSeverity of all instances identified by the input parameter alarmInformationReferenceList are set to 'cleared'. The AlarmInformation.clearUserId and AlarmInformation.clearSystemId of all instances identified are set with values carried by input parameters clearUserId and clearSystemId respectively.
someCleared	It has the same properties as allCleared except that it is applicable to one or more but not all instances identified by the input parameter alarmInformationReferenceList.

6.7.1.6 Exceptions

Name	Properties
operation_failed	**Condition:** the pre-condition is false or the post-condition is false. **Returned Information:** The output parameter status. **Exit state:** Entry state.

6.87 Interface `AlarmIRPNotifications_1`

This specification does not specify methods for IRPManager to detect alarm loss. The use of `alarmId` to detect alarm loss is an arrangement made between `IRPAgent` and IRPManager. This arrangement is outside the scope of this specification. For example, `IRPAgent` may use integer sequence (e.g. 1, 2, 3, 4, 5, …) as `alarmId instances` for its alarms. Based on this knowledge, IRPManager can detect alarm loss. This kind of arrangement may not be possible for all SS.

This specification does not specify how `IRPAgent` can determine if IRPManager has received alarms correctly. Not all SSs provide such capability.

This document does not specify methods for `IRPManager` and `IRPAgent` to recover alarm loss. The only mechanism recommended to deal with alarm loss is the use of `getAlarmList` operation. This document does not specify conditions under which `IRPManager` should invoke this operation.

6.87.1 `notifyNewAlarm` (M)

6.87.1.1 Definition

A new `AlarmInformation` has been added in the `AlarmList`. The subscribed `IRPManager` instances are notified of this fact if the added `AlarmInformation` satisfies the current filter constraint of their subscription.

6.87.1.2 Input Parameters

Parameter Name	Qualifier	Matching Information	Comment
objectClass	M,F	MonitoredEntity.objectClass where the MonitoredEntity is identified by the relation-AlarmedObject-AlarmInformation of the new AlarmInformation.	
objectInstance	M,F	MonitoredEntity.objectInstance where the MonitoredEntity is identified by the relation-AlarmedObject-AlarmInformation of the new AlarmInformation.	
notificationId	M	This carries the semantics of notification identifier.	
eventTime	M,F	AlarmInformation.alarmRaisedTime	
systemDN	C,F	IRPAgent.systemDN where the IRPAgent is related to the AlarmIRP that is related to this AlarmList.	It carries the DN of the IRPAgent.
notificationType	M,F	“notifyNewAlarm”.	
probableCause	M,F	AlarmInformation.probableCause	
perceivedSeverity	M,F	AlarmInformation.perceivedSeverity	
alarmType	M, F	AlarmInformation.eventType	
specificProblem	O	AlarmInformation.specificProblem	
correlatedNotifications	O	The set of CorrelatedNotification related to this AlarmInformation.	
backedUpStatus	O	AlarmInformation.backedUpStatus	
backUpObject	O	MonitoredEntity.objectInstance where the MonitoredEntity is identified by relation-BackUpObject-AlarmInformation of the new AlarmInformation.	It carries the DN of the back up object.
trendIndication	O	AlarmInformation.trendIndication	
thresholdInfo	O	AlarmInformation.thresholdInfo	
stateChangeDefinition	O	AlarmInformation.stateChange	
monitoredAttributes	O	AlarmInformation.monitoredAttributes	
proposedRepairActions	O	AlarmInformaton.proposedRepairActions	
additionalText	O	AlarmInformation.additionalText	
alarmId	M	AlarmInformation.alarmId	

6.87.1.3 Triggering Event

6.87.1.3.1 From-state

`noMatchedAlarm.`

Assertion Name	Definition
noMatchedAlarm	AlarmList does not contain an AlarmInformation that has the following properties: Its matching-criteria-attributes values are identical to that of the newly generated network alarm and It is involved in relation-AlarmObject-AlarmInformation with the same MonitoredEntity as the one identified by the newly generated network alarm.

6.87.1.3.2 To-state

`newAlarmInAlarmList.`

Assertion Name	Definition
newAlarmInAlarmList	AlarmList contains an AlarmInformation holding information conveyed by the newly generated network alarm. This AlarmInformation is involved in relation-AlarmObject-AlarmInformation with the same MonitoredEntity as the one identified by the newly generated network alarm. The following attributes of the AlarmInformation shall be populated with information in the newly generated alarm. alarmId, notificationId, alarmRaisedTime, eventType, probableCause, perceivedSeverity. The following attributes of the same AlarmInformation shall be populated with information in the newly generated alarm if the information is present (in the newly generated alarm) and if the attribute is supported. specificProblem, backedUpStatus, trendIndication, thresholdInfo, stateChangedDefinition, monitoredAttributes, proposedRepairActions, additionalText.

6.87.2 notifyAckStateChanged (M)

6.87.2.1 Definition

The subscribed IRPManager instances are notified regarding changes in alarm Acknowledgement State. The `AlarmInformation` carried in the notification shall satisfy the current filter constraint of the subscription.

The notification shall contain all parameters that are filterable and are present in the original (related) `notifyNewAlarm` notification.

IRPManager or IRPAgent can acknowledge `AlarmInformation` as defined by [9].

6.87.2.2 Input Parameters

Parameter Name	Qualifier	Matching Information	Comment
objectClass	M,F	MonitoredEntity.objectClass where the MonitoredEntity is identified by the relation-AlarmedObject-AlarmInformation of the AlarmInformation.	
objectInstance	M,F	MonitoredEntity.objectInstance where the MonitoredEntity is identified by the relation-AlarmedObject-AlarmInformation of the AlarmInformation.	
notificationId	M	This carries the semantics of notification identifier.	
eventTime	M,F	AlarmInformation.ackTime	
systemDN	C,F	IRPAgent.systemDN	
notificationType	M,F	“notifyAckStateChanged”	
probableCause	M,F	AlarmInformation.probableCause	
perceived Severity	M,F	AlarmInformation.perceivedSeverity	
alarmType	M,F	AlarmInformation.eventType	
alarmId	M	AlarmInformation.alarmId	
ackTime	M	AlarmInformation.ackTime	
ackState	M	AlarmInformation.ackState	
ackUserId	M	AlarmInformation.ackUserId	If this AlarmInformation has been acknowledged by a human operator, than this parameter contains the operator identifier. If it has been acknowledged by a System (EM or NM), than this parameter contains the identifier of the System.
ackSystemId	O	AlarmInformation.ackSystemId	This parameter always contains the identifier of the System (EM or NM) where the acknowledgement request was originated.

6.87.2.3 Triggering Event

6.87.2.3.1 From-state

`alarmInformationExists`.

Assertion Name	Definition
alarmInformationExists	The AlarmInformation exists in AlarmList.

6.8~~7~~.2.3.2 To-state

`alarmAckStateHasChanged.`

Assertion Name	Definition
alarmAckStateHasChanged	The AlarmInformation.ackState of the AlarmInformation identified by from-state assertion alarmInformationExists have been updated. Specifically, the following attributes of the subject AlarmInformation are updated. notificationId, ackTime, ackUserId, ackState, ackSystemId.

6.8~~7~~.3 `notifyClearedAlarm` (M)

6.8~~7~~.3.1 Definition

IRPAgent notifies ~~T~~the subscribed `IRPManager` ~~is notified~~ of alarm clearing if the subject `AlarmInformation` satisfies the optional filter constraint expressed in the `subscribe` operation.

The notification shall contain all parameters that are filterable and are present in the original (related) `notifyNewAlarm` notification.

6.8~~7~~.3.2 Input Parameters

Parameter Name	Qualifier	Matching Information	Comment
objectClass	M,F	MonitoredEntity.objectClass where the MonitoredEntity is identified by the relation-AlarmedObject-AlarmInformation of the new AlarmInformation.	
objectInstance	M,F	MonitoredEntity.objectInstance where the MonitoredEntity is identified by the relation-AlarmedObject-AlarmInformation of the new AlarmInformation.	
notificationId	M	This carries the semantics of notification identifier.	
eventTime	M,F	AlarmInformation.alarmClearedTime	
systemDN	C,F	IRPAgent.systemDN where the IRPAgent is related to the AlarmIRP that is related to this AlarmList.	
notificationType	M,F	“notifyClearedAlarm”	
probableCause	M,F	AlarmInformation.probableCause	
perceivedSeverity	M,F	AlarmInformation.perceivedSeverity	Its value shall indicate Cleared.
alarmType	M,F	AlarmInformation.eventType	
correlated Notifications	O	The set of CorrelatedNotification related to this AlarmInformation.	It contains references to other AlarmInformation instances whose perceivedSeverity levels are Cleared as well. In this way, perceivedSeverity level of multiple AlarmInformation instances can be Cleared by one notification.
clearUserId	O	AlarmInformation.clearUserId	It is present if the AlarmInformation is cleared by the IRPManager using clearAlarms.
clearSystemId	O	AlarmInformation.clearSystemId	It is present if clearUserId is present and if AlarmInformation.clearSystemId contains information.
alarmId	M	AlarmInformation.alarmId	

6.87.3.3 Triggering Event

6.87.3.3.1 From-state

`alarmMatched`AndCleared `OR` ~~`AND alarmCleared`~~ `clearedByIRPManager`.

Assertion Name	Definition
alarmMatchedAndCleared	The matching-criteria-attributes of the newly generated network alarm have values that are identical (matched) with ones in one AlarmInformation in AlarmList and the perceivedSeverity of the matched AlarmInformation is not Cleared AND The perceivedSeverity of the newly generated network alarm is cleared.
~~alarmCleared~~clearedByIRPManager	~~The perceivedSeverity of the newly generated network alarm is Cleared.~~Reception of a valid clearAlarms operation that identifies the subject AlarmInformation instances. This triggering event shall occur regardless of the perceivedSeverity state of the identified AlarmInformation instances.

6.87.3.3.2 To-state

`AlarmInformationCleared_1` `OR AlarmInformationCleared_2`~~`,`~~`.`

Assertion Name	Definition
AlarmInformationCleared_1	Case if From-state is alarmMatchedAndCleared: The following attributes of the subject `AlarmInformation` are updated:~~,~~ `notificationId`, `perceivedSeverity` (updated to `Cleared`), `alarmClearedTime`.
AlarmInformationCleared_2	Case if From-state is clearedByIRPManager: The following attributes of the subject AlarmInformation are updated: notificationId, perceivedSeverity (updated to Cleared), alarmClearedTime, alarmClearedUserId, alarmClearedSystemId.

6.87.4 `notifyAlarmListRebuilt` (M)

6.87.4.1 Definition

The `IRPAgent` or its related `AlarmIRP` maintains an `AlarmList`. They can lose confidence in the integrity of its `AlarmList`. Under this condition, `IRPAgent` or its related `AlarmIRP` or the related `AlarmList` shall invoke `notifyAlarmListRebuilt` notification after the `AlarmList` has been rebuilt.

The `IRPAgent` can also invoke `notifyAlarmListRebuilt` notification indicating that part of the `AlarmList` has been rebuilt. In this case, the notification carries the managed object (MO) instance indicating that the `AlarmList` only have been rebuilt for alarms concerning this MO and its subordinate MOs. Furthermore, this notification indicates that there is no rebuilt going on for superior MOs of this MO.

6.87.4.2 Input Parameters

Parameter Name	Qualifier	Matching Information	Comment
objectClass	M,F	It carries the IRPAgent.objectClass or alternatively, the object class of another MO.	If it carries the IRPAgent.objectClass, then all `AlarmInformation` instances in the AlarmList may have been rebuilt. If it carries the object class of another MO, then all `AlarmInformation` instances of the MO identified by the parameter `objectInstance` and its subordinate MOs may have been rebuilt. The `AlarmInformation` instances not related to the subject MO and its subordinate MOs are not subject to rebuilt.
objectInstance	M,F	It carries the IRPAgent.iRPAgentId or alternatively, the id of another MO.	If objectClass carries the IRPAgent.objectClass, then this parameter carries the RDN of the IRPAgent whose AlarmList has been rebuilt. If objectClass carries the object class of another MO, then this parameter carries the RDN of the MO instance indicating that the AlarmList only have been rebuilt for alarms concerning that MO and its subordinate MOs.
notificationId	M	This carries the semantics of notification identifier.	
eventTime	M,F	It carries the time when the IRPAgent has rebuilt the AlarmList successfully.	
systemDN	C,F	IRPAgent.systemDN where the IRPAgent is related to the AlarmIRP that is related to this AlarmList.	
notificationType	M,F	“notifyAlarmListRebuilt”.	
reason	M	“Agent-NE communication error”, “Agent restarts”, “indeterminate”. Other values can be added.	It carries the reason why the IRPAgent has rebuilt the AlarmList. This may carry different reasons than that carried by the immediate previous notifyPotentialFaultyAlarmList.
alarmListAlignmentRequirement	O (note)	ENUM (alignmentRequired, alignmentNotRequired)	It carries an enumeration of “alignmentRequired” and “alignmentNotRequired”. IRPAgent uses alignmentRequired to indicate that IRPAgent current AL is not identical to the one that could have been built using (a) IRPAgent AL information at the time it emits the immediate previous notifyPotentialFaultyAlarmList() and (b) the notifications (carrying alarm information) emitted after the previously identified notification and before the subject notification. Otherwise, the IRPAgent uses alignmentNotRequired. When this parameter is absent, it implies alignmentRequired.

NOTE: If IRPAgent supports notifyPotentialFaultyAlarmList() notification, it shall support this parameter. If IRPAgent does not support notifyPotentialFaultyAlarmList() notification, it shall not support this parameter.

6.87.4.3 Triggering Event

6.87.4.3.1 From-state

`alarmListRebuilt_0 OR alarmListRebuilt_1.`

Assertion Name	Definition
alarmListRebuilt_0	IRPAgent has cold-started, initialised, re-initialised or rebooted and it has initiated procedure to rebuild its AlarmList.
alarmListRebuilt_1	IRPAgent loses confidence in part or whole of its AlarmList. IRPAgent has initiated procedure to repair its AlarmList.

6.87.4.3.2 To-state

`alarmListRebuilt_2.`

Assertion Name	Definition
alarmListRebuilt_2	IRPAgent rebuilt the whole or part of AlarmList.

6.98 Interface `AlarmIRPNotification_2`

6.98.1 `notifyChangedAlarm` (O)

6.98.1.1 Definition

The subscribed `IRPManager` instances are notified regarding changes in `AlarmInformation` in `AlarmList`. This notification is only triggered by a change in `perceivedSeverity` attribute value (except to the value "Cleared"). The `AlarmInformation` carried in the notification shall satisfy the current filter constraint of the subscription.

The notification shall contain all parameters that are filterable and are present in the original (related) `notifyNewAlarm` notification.

6.98.1.2 Input Parameters

Parameter Name	Qualifier	Matching Information	Comment
objectClass	M,F	MonitoredEntity.objectClass where the MonitoredEntity is identified by the relation-AlarmedObject-AlarmInformation of the new AlarmInformation.	
objectInstance	M,F	MonitoredEntity.objectInstance where the MonitoredEntity is identified by the relation-AlarmedObject-AlarmInformation of the new AlarmInformation.	
notificationId	M	This carries the semantics of notification identifier.	
eventTime	M,F	AlarmInformation.alarmChangedTime	
systemDN	C,F	IRPAgent.systemDN where the IRPAgent is related to the AlarmIRP that is related to this AlarmList.	
notificationType	M,F	"notifyChangedAlarm"	
probableCause	M,F	AlarmInformation.probableCause	
perceivedSeverity	M,F	AlarmInformation.perceivedSeverity	
alarmType	M,F	AlarmInformation.eventType	
alarmId	M	AlarmInformation.alarmId	

6.98.1.3 Triggering Event

6.98.1.3.1 From-state

alarmMatched AND alarmNotCleared AND alarmChanged.

Assertion Name	Definition
alarmMatched	The matching-criteria-attributes of the newly generated network alarm has values that are identical (matches) with ones in one AlarmInformation in AlarmList.
alarmNotCleared	The perceivedSeverity of the newly generated network alarm is not Cleared.
alarmChanged	The perceivedSeverity of the newly generated network alarm and of the matched AlarmInformation are different.

6.98.1.3.2 To-state

`informationUpdate.`

Assertion Name	Definition
informationUpdate	• The AlarmInformation identified in alarmMatched in from-state has been updated according to the following rules : perceivedSeverity is updated; • notificationId is updated; • alarmChangedTime is updated; • ackTime, ackUserId and ackSystemId are updated to contain no information; • ackState is updated to “unacknowledged”;

6.109 Interface `AlarmIRPNotification_3`

~~6.9~~6.10.1 `notifyComments` (O)

~~6.9~~6.10.1.1 Definition

The subscribed `IRPManager` instances are notified regarding to the addition of `Comment,` as a consequence of successful completion of `setComment` operation, in `AlarmInformation` instances in `AlarmList`. The `AlarmInformation` carried in the notification shall satisfy the current filter constraint of the subscription.

The notification shall contain all parameters that are filterable and are present in the original (related) `notifyNewAlarm` notification.

IRPAgent shall support this notification if it supports the operation `setComment`.

~~6.9~~6.10.1.2 Input Parameters

Parameter Name	Qualifier	Matching Information	Comment
objectClass	M,F	MonitoredEntity.objectClass where the MonitoredEntity is identified by the relation-AlarmedObject-AlarmInformation of the AlarmInformation.	
objectInstance	M,F	MonitoredEntity.objectInstance where the MonitoredEntity is identified by the relation-AlarmedObject-AlarmInformation of the AlarmInformation.	
notificationId	M	This carries the semantics of notification identifier.	
eventTime	M,F	AlarmInformation.alarmChangedTime	
systemDN	C,F	IRPAgent.systemDN	
notificationType	M,F	“notifyComments”	
alarmType	M,F	AlarmInformation.eventType	
probableCause	M,F	AlarmInformation.probableCause	
perceived Severity	M,F	AlarmInformation.perceivedSeverity	
comments	M	The set of Comment instances involved in relationship with this AlarmInformation.	
alarmId	M	AlarmInformation.alarmId	

~~6.9~~6.10.1.3 Triggering Events

~~6.9~~6.10.1.3.1 From-state

`alarmInformationExists.`

Assertion Name	Definition
alarmInformationExists	The AlarmInformation is in AlarmList.

~~6.9~~6.10.1.3.2 To-state

`commentInserted.`

Assertion Name	Definition
commentInserted	One Comment has been created and it is involved in a relationship with the AlarmInformation identified by from-state assertion alarmInformationExists. The following attributes of the newly created Comment shall be populated. commentTime (set to setComment operation completion time), commentText, commentUserId and commentSystemId.

~~6.10~~6.11 Interface AlarmIRPNotification_4

~~6.10~~6.11.1 `notifyPotentialFaultyAlarmList` (O)

~~6.10~~6.11.1.1 Definition

The IRPAgent or its related AlarmIRP maintains an AlarmList. They can lose confidence in the integrity of its AlarmList. Under this condition, IRPAgent or its related AlarmIRP or the related AlarmList shall invoke notifyPotentialFaultyAlarmList. They then can begin to rebuild the faulty AlarmList, if found necessary. After the successful rebuilt or the discovery that rebuilt is not necessary, they shall invoke notifyAlarmListRebuilt notification.

This notification can identify a set of AlarmInformation that is potentially faulty or unreliable. This identification is done in the following way. If the MOI of an AlarmInformation is the same or is a subordinate to the MOI carried in the notification, then the AlarmInformation may be faulty or unreliable.

This notification can identify all the AlarmInformation instances of the AlarmList that are potentially faulty or unreliable. In this case, the notification shall carry a MOI identifying the IRPAgent.

The IRPManager behavior, on reception of this notifyPotentialFaultyAlarmList notification, is not specified. The IRPManager behavior is considered not essential for the specification of the interface itself. However, the following are recommended actions the IRPManager should take, in case it receives this notification.

1. The IRPManager should not perform any task requiring the integrity of the AlarmInformation identified as faulty or unreliable by the subject notification.

2. The IRPManager should not invoke operations that require integrity of the AlarmList such as getAlarmList., acknolwedgeAlarms operations.

~~6.10~~6.11.1.2 Input Parameters

Parameter Name	Qualifier	Matching Information	Comment
objectClass	M,F	It carries the class of the IRPAgent or alternatively, the class of another MO.	If it carries the IRPAgent.objectClass, then all AlarmInformation instances in the AlarmList may not be reliable. If it carries the object class of another MO, then all AlarmInformation instances of the MO identified by the parameter objectInstance and its subordinate MOs may not be reliable. The AlarmInformation instances not related to the subject MO and its subordinate MOs are reliable.
objectInstance	M,F	It carries the DN of the IRPAgent or alternatively, the DN of another MO.	If objectClass carries the IRPAgent.objectClass, then this parameter carries the DN of the IRPAgent. If objectClass carries the object class of another MO, then this parameter carries the DN of the MO instance.
notificationId	M	This carries the semantics of notification identifier.	
eventTime	M,F	It carries the time when the IRPAgent has lost confidence of its AlarmList content.	
systemDN	C,F	IRPAgent.systemDN where the IRPAgent is related to the AlarmIRP that is related to this AlarmList.	
notificationType	M,F	“notifyPotentialFaultyAlarmList” .	
reason	M	“Agent-NE communication error”, “Agent restarts”, “indeterminate”. Other values can be added.	It carries the reason why the IRPAgent has to rebuild its AlarmList.

~~6.10~~6.11.1.3 Triggering Event

~~6.10~~6.11.1.3.1 From-state

`faultyAlarmListDetected`.

Assertion Name	Definition
faultyAlarmListDetected	IRPAgent detects faults in part or whole of its AlarmList.

~~6.10~~6.11.1.3.2 To-state

`faultyAlarmList`

Assertion Name	Definition
faultyAlarmList	IRPAgent initiates the AlarmList rebuild process.

Annex A (normative): Event Types

This appendix lists and explains event types used by this document.

Event type is defined in 3GPP TS 32.302 [5]. The table below lists some of the event types referred to in this document.

Notification IRP: Information Service [5] defines a parameter called `notificationType` that shall be present in all notification. This document defines a parameter called `alarmType` that shall be present in all notifications carrying alarm information. Examples of the `notificationType` are "notification of new alarm", "notification of AlarmList rebuilt", "notification of alarm cleared", etc. Examples of the `alarmType` are the event types defined in table below.

This document also defines an attribute of `AlarmInformation` called `eventType`. The mapping of this `eventType` (internal attribute and not visible to `IRPManager`) to `notificationType` or `alarmType` (both visible to `IRPManager`) is defined in relevant sections of this document. The choice of using "`eventType`" is to keep the list of attributes of `AlarmList` unchanged (compared to Release 99). One can replace this `eventType` with two attributes, called `notificationType` and `alarmType` so that mapping of these two attributes to the externally visible parameters of the same name will be straight-forward.

It is noted that the `AlarmInformation.eventType` can capture more information than the ITU-T defined event types [2]. One example is "notification of alarm list rebuilt".

It is noted that the mapping of the IS `notificationType` and `alarmType` to CMIP's event type or CORBA `event_name` or other fields are specified in the respective SS documents.

TableA.1: Event Types

Event Types	Explanation
Communications Alarm	An alarm of this type is associated with the procedure and/or process required conveying information from one point to another (ITU-T Recommendation X.733 [2]).
Processing Error Alarm	An alarm of this type is associated with a software or processing fault (ITU-T Recommendation X.733 [2]).
Environmental Alarm	An alarm of this type is associated with a condition related to an enclosure in which the equipment resides (ITU-T Recommendation X.733 [2]).
Quality of Service Alarm	An alarm of this type is associated with degradation in the quality of a service (ITU-T Recommendation X.733 [2]).
Equipment Alarm	An alarm of this type is associated with an equipment fault (ITU-T Recommendation X.733 [2]).

Annex B (normative): Probable Causes

This appendix lists probable causes and their corresponding event types.

Sources of these probable causes are ITU-T Recommendation M.3100 [11], ITU-T Recommendation X.721 [3], ITU-T Recommendation X.733 [2], ITU-T Recommendation X.736 [15] and GSM 12.11 [4].

The list may be extended in the future, e.g. with UMTS-specific probable causes.

Table B.1: Probable Causes from ITU-T Recommendation M.3100 [11]

M.3100 Probable cause	Event type
Indeterminate	Unknown
Alarm Indication Signal (AIS)	Communications
Call Setup Failure	Communications
Degraded Signal	Communications
Far End Receiver Failure (FERF)	Communications
Framing Error	Communications
Loss Of Frame (LOF)	Communications
Loss Of Pointer (LOP)	Communications
Loss Of Signal (LOS)	Communications
Payload Type Mismatch	Communications
Transmission Error	Communications
Remote Alarm Interface	Communications
Excessive Bit Error Rate (EBER)	Communications
Path Trace Mismatch	Communications
Unavailable	Communications
Signal Label Mismatch	Communications
Loss Of Multi Frame	Communications
Back Plane Failure	Equipment
Data Set Problem	Equipment
Equipment Identifier Duplication	Equipment
External IF Device Problem	Equipment
Line Card Problem	Equipment
Multiplexer Problem	Equipment
NE Identifier Duplication	Equipment
Power Problem	Equipment
Processor Problem	Equipment
Protection Path Failure	Equipment
Receiver Failure	Equipment
Replaceable Unit Missing	Equipment
Replaceable Unit Type Mismatch	Equipment
Synchronisation Source Mismatch	Equipment
Terminal Problem	Equipment
Timing Problem	Equipment
Transmitter Failure	Equipment
Trunk Card Problem	Equipment
Replaceable Unit Problem	Equipment
Air Compressor Failure	Environmental
Air Conditioning Failure	Environmental
Air Dryer Failure	Environmental
Battery Discharging	Environmental
Battery Failure	Environmental
Commercial Power Failure	Environmental
Cooling Fan Failure	Environmental
Engine Failure	Environmental
Fire Detector Failure	Environmental
Fuse Failure	Environmental
Generator Failure	Environmental
Low Battery Threshold	Environmental
Pump Failure	Environmental
Rectifier Failure	Environmental
Rectifier High Voltage	Environmental
Rectifier Low F Voltage	Environmental
Ventilation System Failure	Environmental
Enclosure Door Open	Environmental
Explosive Gas	Environmental
Fire	Environmental
Flood	Environmental
High Humidity	Environmental
High Temperature	Environmental
High Wind	Environmental
Ice Build Up	Environmental
Intrusion Detection	Environmental

M.3100 Probable cause	Event type
Low Fuel	Environmental
Low Humidity	Environmental
Low Cable Pressure	Environmental
Low Temperature	Environmental
Low Water	Environmental
Smoke	Environmental
Toxic Gas	Environmental
Storage Capacity Problem	Processing error
Memory Mismatch	Processing error
Corrupt Data	Processing error
Out Of CPU Cycles	Processing error
Software Environment Problem	Processing error
Software Download Failure	Processing error

Table B.2: Probable Causes from ITU-T Recommendation X.721 [3] / ITU-T Recommendation X.733 [2]

X.733 Probable Cause	Event type
Adapter Error	Equipment
Application Subsystem Failure	Processing error
Bandwidth Reduction	Quality of service
Call Establishment Error	Communications
Communication Protocol Error	Communications
Communication Subsystem Failure	Communications
Configuration or Customizing Error	Processing error
Congestion	Quality of service
Corrupt Data	Processing error
CPU Cycles Limit Exceeded	Processing error
Data Set or Modem Error	Equipment
Degraded Signal	Communications
DTE-DCE Interface Error	Communications
Enclosure Door Open	Environmental
Equipment Malfunction	Equipment
Excessive Vibration	Environmental
File Error	Processing error
Fire Detected	Environmental
Flood Detected	Environmental
Framing Error	Communications
Heating or Ventilation or Cooling System Problem	Environmental
Humidity Unacceptable	Environmental
Input/Output Device Error	Equipment
Input Device Error	Equipment
LAN Error	Communications
Leak Detection	Environmental
Local Node Transmission Error	Communications
Loss of Frame	Communications
Loss of Signal	Communications
Material Supply Exhausted	Environmental
Multiplexer Problem	Equipment
Out of Memory	Processing error
Output Device Error	Equipment
Performance Degraded	Quality of service
Power Problem	Equipment
Pressure Unacceptable	Environmental
Processor Problem	Equipment
Pump Failure	Environmental
Queue Size Exceeded	Quality of service
Receive Failure	Equipment
Receiver Failure	Equipment
Remote Node Transmission Error	Communications
Resource at or Nearing Capacity	Quality of service
Response Time Excessive	Quality of service
Re-transmission Rate Excessive	Quality of service
Software Error	Processing error
Software Program Abnormally Terminated	Processing error
Software Program Error	Processing error
Storage Capacity Problem	Processing error
Temperature Unacceptable	Environmental
Threshold Crossed	Quality of service
Timing Problem	Equipment
Toxic Leak Detected	Environmental
Transmit Failure	Equipment
Transmitter Failure	Equipment
Underlying Resource Unavailable	Processing error
Version Mismatch	Processing error

Table B.3: Probable Causes from GSM 12.11 [4]

GSM 12.11 Probable Cause	Event Type
A-bis to BTS interface failure	Equipment
A-bis to TRX interface failure	Equipment
Antenna problem	Equipment
Battery breakdown	Equipment
Battery charging fault	Equipment
Clock synchronisation problem	Equipment
Combiner problem	Equipment
Disk problem	Equipment
Equipment failure	Equipment
Excessive receiver temperature	Equipment
Excessive transmitter output power	Equipment
Excessive transmitter temperature	Equipment
Frequency hopping degraded	Equipment
Frequency hopping failure	Equipment
Frequency redefinition failed	Equipment
Line interface failure	Equipment
Link failure	Equipment
Loss of synchronisation	Equipment
Lost redundancy	Equipment
Mains breakdown with battery back-up	Equipment
Mains breakdown without battery back-up	Equipment
Power supply failure	Equipment
Receiver antenna fault	Equipment
Receiver Failure	Equipment
Receiver multicoupler failure	Equipment
Reduced transmitter output power	Equipment
Signal quality evaluation fault	Equipment
Timeslot hardware failure	Equipment
Transceiver problem	Equipment
Transcoder problem	Equipment
Transcoder or rate adapter problem	Equipment
Transmitter antenna failure	Equipment
Transmitter antenna not adjusted	Equipment
Transmitter failure	Equipment
Transmitter low voltage or current	Equipment
Transmitter off frequency	Equipment
Database inconsistency	Processing error
File system call unsuccessful	Processing error
Input parameter out of range	Processing error
Invalid parameter	Processing error
Invalid pointer	Processing error
Message not expected	Processing error
Message not initialised	Processing error
Message out of sequence	Processing error
System call unsuccessful	Processing error
Timeout expired	Processing error
Variable out of range	Processing error
Watch dog timer expired	Processing error
Cooling system failure	Environmental
External equipment failure	Environmental
External power supply failure	Environmental
External transmission device failure	Environmental
Fan failure	Environmental
High humidity	Environmental
High temperature	Environmental
Intrusion detected	Environmental
Low humidity	Environmental
Low temperature	Environmental
Smoke detected	Environmental
Excessive Error Rate	Quality of service
Reduced alarm reporting	Quality of service
Reduced event reporting	Quality of service

GSM 12.11 Probable Cause	Event Type
Reduced logging capability	Quality of service
System resources overload	Quality of service
Broadcast channel failure	Communications
Connection establishment error	Communications
Invalid message received	Communications
Invalid MSU received	Communications
LAPD link protocol failure	Communications
Local alarm indication	Communications
Remote alarm indication	Communications
Routing failure	Communications
SS7 protocol failure	Communications
Transmission error	Communications

Table 20 identifies probable causes that are defined by more than one standard. This is for information only.

Table B.4: Duplicated Probable Causes

Duplicated Probable Cause	GSM 12.11	X.721 X.733	M.3100	Event Type
Call Establishment Failure (X.721/X.733) Call Setup Failure (M.3100)		X	X	Communications
Degraded Signal		X	X	Communications
Framing Error		X	X	Communications
Loss of Frame		X	X	Communications
Loss of Signal		X	X	Communications
Equipment Failure (GSM 12.11) Equipment Malfunction (X.721/X.733)	X	X		Equipment
Multiplexer Problem		X	X	Equipment
Power Problem		X	X	Equipment
Processor Problem		X	X	Equipment
Receiver Failure	X	X	X	Equipment
Timing Problem		X	X	Equipment
Transmitter Failure	X	X	X	Equipment
Enclosure Door Open		X	X	Environmental
Fan Failure (GSM 12.11) Cooling Fan Failure (M.3100)	X		X	Environmental
Fire Detected (X.721/X.733) Fire (M.3100)		X	X	Environmental
Flood Detected (X.721/X.733) Flood (M.3100)		X	X	Environmental
High Humidity	X		X	Environmental
High Temperature	X		X	Environmental
Intrusion Detected (GSM 12.11) Intrusion Detection (X.736/M.3100)	X		X	Environmental
Low Humidity	X		X	Environmental
Low Temperature	X		X	Environmental
Pump Failure		X	X	Environmental
Smoke Detected (GSM 12.11) Smoke (M.3100)	X		X	Environmental
Storage Capacity Problem		X	X	Processing Error
Excessive Bit Error Rate (M.3100) Excessive Error Rate (GSM12.11)	X		X	
Corrupt Data		X	X	Processing Error

Annex C (informative): Examples of using notifyChangedAlarm

This annex describes a number of valid and invalid interactions governing the case when IRPAgent is reporting a specific fault of a particular network resource whose alarm severity level changes from, e.g. “Critical” to “Minor” and then to “Cleared”.

In the following examples:

ni is notificationId,

moc is managedObjectClass,

moi is managedObjectInstance,

et is eventType,

pc is probableCause,

sp is specificProblem,

ps is perceivedSeverity and

ai is alarmId.

EXAMPLE 1: Valid sequence 1 to support the hypothetical case:

(1) NotifyNewAlarm

(ni=1, ai=X, moc=A, moi=B, et=C, pc=D, sp=E, ps=Critical)

(2) NotifyChangedAlarm

(ni=2, ai=X, moc=A, moi=B, et=C, pc=D, sp=E, ps=Minor)

(3) NotifyClearedAlarm

(ni=3, ai=X, moc=A, moi=B, et=C, pc=D, sp=E, ps=Cleared)

EXAMPLE 2: Valid sequence 2 to support the hypothetical case (assuming that the alarm with “ai=X” is acknowledged after either (1) or (2), but before (3)):

(1) NotifyNewAlarm

(ni=1, ai=X, moc=A, moi=B, et=C, pc=D, sp=E, ps=Critical)

(2) NotifyClearedAlarm

(ni=2, ai=X, moc=A, moi=B, et=C, pc=D, sp=E, ps=Cleared)

(3) NotifyNewAlarm

(ni=3, ai=Y, moc=A, moi=B, et=C, pc=D, sp=E, ps=Minor)

(4) NotifyClearedAlarm

(ni=4, ai=Y, moc=A, moi=B, et=C, pc=D, sp=E, ps=Cleared)

EXAMPLE 3: Invalid sequence 1 to support the hypothetical case:

```
(1) NotifyNewAlarm
    (ni=1, ai=X, moc=A, moi=B, et=C, pc=D, sp=E, ps=Critical)
(2) NotifyChangedAlarm
    (ni=2, ai=Y, moc=A, moi=B, et=C, pc=D, sp=E, ps=Minor)
(3) NotifyClearedAlarm
    (ni=3, ai=Y, moc=A, moi=B, et=C, pc=D, sp=E, ps=Cleared)
```

Interaction (2) is illegal since it uses a different `ai` for the same alarm. It should use `ai=X` as in interaction (1).

EXAMPLE 4: Invalid sequence 2 to support the hypothetical case:

```
(1)  NotifyNewAlarm
   (ni=1, ai=X, moc=A, moi=B, et=C, pc=D, sp=E, ps=Critical)
(2)  NotifyNewAlarm
   (ni=2, ai=X, moc=A, moi=B, et=C, pc=D, sp=E, ps=Minor)
```

Interaction (2) is illegal since it invokes notifyNewAlarm using same `ai` value. It should use `notifyChangedAlarm` with the same `ai` value.

Annex D (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
Mar 2000	S_07	SP-000012	--	--	Approved at TSG SA #7 and placed under Change Control	2.0.0	3.0.0
Mar 2000	--	--	--	--	Cosmetic	3.0.0	3.0.1
Jun 2000	S_08	SP-000250	004	--	Split of TS - Part 2: Alarm Integration Reference Point (IRP): Information Service (IS)	3.0.1	3.1.0
Sep 2000	--	--	--	--	Cosmetic	3.1.0	3.1.1
Sep 2000	S_09	SP-000438	001	--	Correction of qualifier for SystemDN	3.1.1	3.2.0
Sep 2000	S_09	SP-000438	002	--	Addition of a missing constraint in acknowledgeAlarm operation	3.1.1	3.2.0
Dec 2000	S_10	SP-000520	003	--	Incorrect modifiable attributes	3.2.0	3.3.0
Dec 2000	S_10	SP-000520	004	--	Add acknowledgement information to getAlarmList result	3.2.0	3.3.0
Dec 2000	S_10	SP-000520	005	--	Identification of valid Event Types and Extended Event Types within Notifications	3.2.0	3.3.0
Dec 2000	S_10	SP-000520	006	--	A cleared Alarm shall be given perceived severity "Cleared" and nothing else	3.2.0	3.3.0
Dec 2000	S_10	SP-000520	007	--	Inconsistent behaviour for cleared not yet acknowledged alarms	3.2.0	3.3.0
Jun 2001	S_12	SP-010282	008	--	Alarm IRP: IS Rel4 - Addition of feature	3.3.1	4.0.0
Sep 2001	S_13	SP-010474	009	--	Definition of thresholdInfo in Alarm IRP: IS	4.0.0	4.1.0
Dec 2001	S_14	SP-010639	010	--	Correction of notifyChangedAlarm example #2	4.1.0	4.2.0
Dec 2001	S_14	SP-010639	011	--	Update of notificationId missing in To-state of notifyClearedAlarm	4.1.0	4.2.0
Mar 2002	S_15	SP-020028	012	--	Addition of "perceivedSeverity" as parameter to "acknowledgeAlarms operation" (IS)	4.2.0	4.3.0
Mar 2002	S_15	SP-020039	013	--	Addition of parameter in Alarm List Rebuilt notification	4.2.0	4.3.0
Mar 2002	S_15	SP-020039	014	--	Addition of new notification notifyPotentialFaultyAlarmList	4.2.0	4.3.0
Mar 2002	S_15	SP-020039	015	--	Additional trigger event for notifyAlarmListRebuilt	4.2.0	4.3.0
Mar 2002	S_15	--	--	--	Automatic upgrade to Rel-5 (no Rel-5 CR)	4.3.0	5.0.0

3GPP TSG-SA5 (Telecom Management) **S5-026492**
Meeting #29, Beijing, PRC, June 2002

CR-Form-v5

CHANGE REQUEST

⌘ **32.111-3** **CR** **018** ⌘ **rev** **-** ⌘ Current version: **5.0.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE ☐ Radio Access Network **X** Core Network **X**

Title:	⌘	Add clearAlarm and other updates
Source:	⌘	S5
Work item code:	⌘	OAM-NIM
Date:	⌘	28/06/2002
Category:	⌘	**B**
Release:	⌘	*REL-5*

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(addition of feature),*
C *(functional modification of feature)*
D *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
REL-4 (Release 4)
REL-5 (Release 5)

Reason for change:	⌘	Add clearAlarms to support ADMC feature (as defined in current 32.111-1). The sub-clause "Scope" misses an agreed sentence that identifies the IS (Information Service) specification to which this specification is related.
Summary of change:	⌘	Add a new clear_alarms method. Add the sentence in sub-clause scope to specify the IS specification to which this specification is related. Add to "Scope" the sentence that identifies the IS specification to which this specification is related. Remove IDL statement related to IRPVersion.
Consequences if not approved:	⌘	This solution set would not satisfy ADMC requirement. The SS would not specify its SS-IS relation in Scope sub-clause.

Clauses affected:	⌘	1, 2, Table 1, 5.2, Table 2, Table 3, Table 7, (new) Table 10, Table 12, Annex A.
Other specs affected:	⌘	☐ Other core specifications ⌘ ☐ Test specifications ☐ O&M Specifications
Other comments:	⌘	This "child" CR can only be approved if its "grand-parent" 32.111-1CR004 (S5-026491) and its "parent" 32.111-2CR017 (S5-026490) are approved.

1 Scope

The present document specifies the CORBA Solution Set (SS) for the IRP whose semantics is specified in Alarm IRP: Information Service (IS) (3G TS 32.111-2 [6]).

Clause 1 to 3 provides background information. Clause 4 provides key architectural features supporting the SS. Clause 5 defines the mapping of operations, notification, parameters and attributes defined in IS to their SS equivalents. Clause 6 describes the notification interface containing the push method. Annex A contains the IDL specification.

This Solution Set specification is related to 3G TS 32.111-2 V5.0.X.

2 References

The following documents contain provisions, which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.

[1] OMG TC Document telecom/98-11-01: “OMG Notification Service”.

[2] OMG CORBA Services: “Common Object Services Specification, Update: November 22, 1996” (Clause 4 contains the Event Service specification).

[3] 3GPP TS 32.300: “Name Convention for Managed Objects”.

[4] 3GPP TS 32.302: “Notification IRP: Information Service”.

[5] 3GPP TS 32.303: “Notification IRP: CORBA Solution Set”.

[6] 3GPP TS 32.111-2: “Alarm Integration Reference Point: Information Service”.

[7] 3GPP TS 32.312: “Generic IRP Management: Information Service”.

[8] 3GPP TS 32.311: "Generic IRP Management: Requirements".

3 Definitions and abbreviations

3.1 Definitions

In addition to the terms and definitions defined in TS 32.111-2 [6], there are no additional definitions applicable to the present document.

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

CORBA	Common Object Request Broker Architecture
IDL	Interface Definition Language
IRP	Integration Reference Point
MOC	Managed Object Class
MOI	Managed Object Instance
NE	Network Element

OMG	Object Management Group
TMN	Telecommunications Management Network
UML	Unified Model Language

3.3 IRP document version number string

The IRP document version number (sometimes called "IRP version" or "version number") string is used to identify this specification. ~~The string is derived using the following rule.~~The definition of "IRP document version number string" in 3G TS 32.311 [8] provides the rule to derive such a string.

~~Take the 3GPP document number on the front page of this specification, such as "3GPP TS 32.106-3 V3.2.0 (2000-12)". Discard the leading "3GPP TS ". Discard all characters after and including the last period. Eliminate leading and trailing spaces. Reduce multiple consecutive spaces with one space. Express the resultant in a string. Capitalised the string. For example, if the 3GPP document version number is "3GPP TS 32.106-3 V3.2.0 (2000-12)", then the IRP document version number shall be "32.106 V3.2".~~

This string is used for the return value of `get_alarm_IRP_versions()`. It is used as return value of `get_notification_categories()` if the Notification IRP supports the emission of notifications defined by this Alarm IRP version. It is also used in the domain_name attribute of a structured event carrying alarm information defined by this Alarm IRP version.

~~`This string is returned in getAlarmIRPVersion method and is carried in the first field of the notification header of all notifications related to alarm IRP.`~~

4 Architectural Features

The overall architectural feature of Alarm IRP is specified in 3G TS 32.111-2 [6]. This clause specifies features that are specific to the CORBA SS.

4.1 Notification Services

In implementations of CORBA SS, IRPAgent conveys Alarm Information to IRPManager via OMG Notification Service (OMG Notification Service [1]).

OMG Event Service [2] provides event routing and distribution capabilities. OMG Notification Service provides, in addition to Event Service, event filtering and Quality Of Service (QOS) as well.

A necessary and sufficient sub set of OMG Notification Services shall be used to support `AlarmIRPNotifications` notifications as specified in 3G TS 32.111-2 [6].

4.2 Push and Pull Style

OMG Notification Service defines two styles of interaction. One is called push style. In this style, IRPAgent pushes notifications to IRPManager as soon as they are available. The other is called pull style. In this style, IRPAgent keeps the notifications till IRPManager requests for them.

This CORBA SS specifies that support of Push style is Mandatory (M) and that support of Pull style is Optional (O).

4.3 Support multiple notifications in one `push` operation

For efficiency reasons, IRPAgent may send multiple notifications using one single `push` operation. To pack multiple notifications into one `push` operation, IRPAgent may wait and not invoke the `push` operation as soon as notifications are available. To avoid IRPAgent to wait for an extended period of time that is objectionable to IRPManager, IRPAgent shall implement an IRPAgent wide timer configurable by administrator. On expiration of this timer, IRPAgent shall invoke `push` if there is at least one notification to be conveyed to IRPManager. This timer is re-started after each `push` invocation.

4.4 Filter

IRPAgent shall optionally support alarm filtering based on IRPManager's supplied alarm filter constraints (e.g., as parameter in `subscribe`() of 3G TS 32.302 [4]. Alarm filtering can be applied in the following cases:

- It is applicable to alarms emitted by IRPAgent via `AlarmIRPNotifications`. IRPManager supplies alarm filter constraint via the `subscribe` method. This filter is effective during the period of subscription.
- It is applicable to alarms returned by IRPAgent via the `out` parameter of `get_alarm_list` method. IRPManager supplies alarm filter constraint via the `get_alarm_list` method. This filter is effective only for this method invocation.
- It is applicable to the calculation of alarm counts returned by IRPAgent via the `out` parameters of `get_alarm_count` method. IRPManager supplies alarm filter constraint via the `get_alarm_count` method. This filter is effective only for this method invocation.

This SS shall use of filter constraint grammar specified by reference OMG Notification Service [1]. The name of the grammar is called "`EXTENDED_TCL`". See clause 2.4, Default Filter Constraint Language in OMG Notification Service [1]. This SS shall use this grammar only.

5 Mapping

5.1 Operation and Notification mapping

Alarm IRP: IS 3G TS 32.111-2 [6] defines semantics of operation and notification visible across the Alarm IRP. Table 1 indicates mapping of these operations and notifications to their equivalents defined in this SS.

Table 1: Mapping from IS Notification/Operation to SS equivalents

IS Operation/ notification 3G TS 32.111-2 [613]	SS Method	Qualifier
acknowledgeAlarms	acknowledge_alarms	M
unacknowledgeAlarms	unacknowledge_alarms	O
getAlarmList	get_alarm_list	M
getIRPVersion	get_alarm_IRP_versions	M
getAlarmCount	get_alarm_count	O
setComment	comment_alarms	O
clearAlarms	clear_alarms	O
getOperationProfile (note)	get_alarm_IRP_operations_profile	O
getNotificationProfile (note)	get_alarm_IRP_notification_profile	O
notifyNewAlarm	push_structured_event Note that OMG Notification Service OMG Notification Service [1] defines this method. See clause 6.1	M
notifyClearedAlarm	push_structured_event See clause 6.1	M
notifyChangedAlarm	push_structured_event See clause 6.1	M
notifyAckStateChanged	push_structured_event See clause 6.1	M
notifyAlarmListRebuilt	push_structured_event See clause 6.1	M
notifyComments	push_structured_event See clause 6.1	O
NOTE: This operation is of ManagedGenericIRP IOC specified in [7]. The AlarmIRP IOC of [6] inherits from it.		

5.2 Operation parameter mapping

Reference 3G TS 32.111-2 [6] defines semantics of parameters carried in operations across the Alarm IRP. The following set of tables indicates the mapping of these parameters, as per operation, to their equivalents defined in this SS.

Table 2: Mapping from IS `acknowledgeAlarms` parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
alarmInformationAndSeverity ReferenceList	AlarmIRPConstDefs::AlarmInformationIdAndSevSeq alarm_information_id_and_sev_list Note: perceivedSeverity is optional { alarmId - Mandatory; perceivedSeverity - Optional }	M
ackUserId	string ack_user_id	M
ackSystemId	string ack_system_id	O
bad-AlarmInformation-ReferenceList	AlarmIRPConstDefs::BadAcknowledgeAlarmInfoSeq bad_ack_alarm_info_list	M
status	ManagedGenericIRPConstDefs::Signal Exceptions: AcknowledgeAlarms, ManagedGenericIRPSystem::ParameterNotSupported, ManagedGenericIRPSystem::InvalidParameter	M

Table 3: Mapping from IS `unacknowledgeAlarms` parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
alarm-InformationReferenceList	AlarmIRPConstDefs::AlarmInformationIdSeq alarm_information_id_list	M
ackUserId	string ack_user_id	M
ackSystemId	string ack_system_id	O
badAlarm-Information-ReferenceList	AlarmIRPConstDefs:: BadAlarmInformationIdSeq bad_alarm_information_id_list	M
status	ManagedGenericIRPConstDefs::Signal Exceptions: UnacknowledgeAlarms, ManagedGenericIRPSystem::OperationNotSupported, ManagedGenericIRPSystem::ParameterNotSupported, ManagedGenericIRPSystem::InvalidParameter	M

Table 4: Mapping from IS `getAlarmList` parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
alarmAckState, filter	string filter	O
alarmInformation List	Return value of type AlarmIRPConstDefs::AlarmInformationSeq	M
status	Exceptions: GetAlarmList, ManagedGenericIRPSystem::ParameterNotSupported, ManagedGenericIRPSystem::InvalidParameter	M

Table 5: Mapping from IS `getAlarmCount` parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
alarmAckState, filter	string filter	O
criticalCount, majorCount, minorCount, warningCount, indeterminateCount,clearedCount	long critical_count, long major_count, long minor_count, long warning_count, long indeterminate_count, long cleared_count	M
status	Exceptions: GetAlarmCount, ManagedGenericIRPSystem::OperationNotSupported, ManagedGenericIRPSystem::ParameterNotSupported, ManagedGenericIRPSystem::InvalidParameter	M

Table 6: Mapping from IS `getIRPVersion` parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
versionNumberSet	Return value of type ManagedGenericIRPConstDefs::VersionNumberSet	M
status	Exceptions: GetAlarmIRPVersions	M

Table 7: Mapping from IS `setComment` parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
AlarmInformation-ReferenceList	AlarmIRPConstDefs::AlarmInformationIdSeq alarm_information_id_list	M
commentUserId	string comment_user_id	M
commentSystemId	string comment_system_id	O
commentText	string comment_text	M
badAlarmInformationReferenceList	AlarmIRPConstDefs::BadAlarmInformationIdSeq bad_alarm_information_id_list	M
status	ManagedGenericIRPConstDefs::Signal Exceptions: CommentAlarms, ManagedGenericIRPSystem::OperationNotSupported, ManagedGenericIRPSystem::ParameterNotSupported ManagedGenericIRPSystem::InvalidParameter	M

Table 8: Mapping from IS `getOperationProfile` parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
irpVersion	ManagedGenericIRPConstDefs::VersionNumber alarm_irp_version	M
operationNameProfile, operationParameterProfile	Return value of type ManagedGenericIRPConstDefs::MethodList	M
status	Exceptions: GetAlarmIRPOperationsProfile, ManagedGenericIRPSystem::OperationNotSupported, ManagedGenericIRPSystem::InvalidParameter	M

Table 9: Mapping from IS `getNotificationProfile` parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
irpVersion	ManagedGenericIRPConstDefs::VersionNumber alarm_irp_version	M
notificationNameProfile, notificationParameterProfile	Return value of type ManagedGenericIRPConstDefs::MethodList	M
status	Exceptions: GetAlarmIRPNotificationProfile, ManagedGenericIRPSystem::OperationNotSupported, ManagedGenericIRPSystem::InvalidParameter	M

Table 10: Mapping from IS `clearAlarms` parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
alarmInformationReferenceList	AlarmIRPConstDefs::AlarmInformationIdSeq alarm_information_id_list	M
clearUserId	string clear_user_id	M
clearSystemId	string clear_system_id	O
badAlarmInformationReferenceList	AlarmIRPConstDefs:: BadAlarmInformationIdSeq bad_alarm_information_id_list	M
status	ManagedGenericIRPConstDefs::Signal Exceptions: ClearAlarms, ManagedGenericIRPSystem::OperationNotSupported, ManagedGenericIRPSystem::ParameterNotSupported, ManagedGenericIRPSystem::InvalidParameter	M

5.3 Notification parameter mapping

Reference 3G TS 32.111-2 [6] defines semantics of parameters carried in notifications. The following tables indicate the mapping of these parameters to their OMG CORBA Structured Event (defined in OMG Notification Service [1]) equivalents. The composition of OMG Structured Event, as defined in the OMG Notification Service [1], is:

```
Header
      Fixed Header
           domain_name
           type_name
           event_name
      Variable Header
Body
      filterable_body_fields
      remaining_body
```

The following tables list all OMG Structured Event attributes in the second column. The first column identifies the Alarm IRP: IS [6] defined notification parameters.

Table 10: Mapping for `notifyNewAlarm`

IS Parameters	OMG CORBA Structured Event attribute	Qualifier	Comment
There is no corresponding SS attribute.	domain_name		It carries the IRP document version number string. See sub-clause 3.3. It indicates the syntax and semantics of the Structured Event as defined by this specification.
notificationType	type_name	M	This is the NOTIFY_FM_NEW_ALARM of interface NotificationType of module AlarmIRPConstDefs.
alarmType	event_name	M	It identifies one of the following: communications alarm, processing error alarm, environmental alarm, quality of service alarm and equipment alarm. It is a string defined by interface AlarmType of module AlarmIRPConstDefs.
There is no corresponding SS attribute.	variable Header		
objectClass, objectInstance	One NV pair of filterable_ body_fields	M	NV stands for name-value pair. Order arrangement of NV pairs is not significant. The name of NV-pair is always encoded in string. Name of NV pair is the MANAGED_OBJECT_INSTANCE of interface AttributeNameValue of module NotificationIRPConstDefs. Value of NV pair is a string.
notification Id	One NV pair of filterable_ body_fields	M	Name of NV pair is the NOTIFICATION_ID of interface AttributeNameValue of module NotificationIRPConstDefs. Value of NV pair is a long.
eventTime	One NV pair of filterable_ body_fields	M	Name of NV pair is the EVENT_TIME of interface AttributeNameValue of module NotificationIRPConstDefs. Value of NV pair is a IRPTime of module ManagedGenericIRPConstDefs.
systemDN	One NV pair of filterable_ body_fields	M	Name of NV pair is the SYSTEM_DN of interface AttributeNameValue of module NotificationIRPConstDefs. Value of NV pair is a string.
probableCause	One NV pair of filterable_ body_fields	M	Name of NV pair is the PROBABLE_CAUSE of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a short defined by interface ProbableCause of module AlarmIRPConstDefs.
perceivedSeverity	One NV pair of filterable_ body_fields	M	Name of NV pair is the PERCEIVED_SEVERITY of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a short defined by interface PerceivedSeverity of module AlarmIRPConstDefs.
specificProblem	One NV pair of filterable_ body_fields	O	Name of NV pair is the SPECIFIC_PROBLEM of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a string.
correlatedNotifications	One NV pair of filterable_ body_fields	O	Name of NV pair is the CORRELATED_NOTIFICATIONS of interface AttributeNameValue. Value of NV pair is a CorrelatedNotificationSetType of module AlarmIRPConstDefs.
backedUpStatus	One NV pair of filterable_ body_fields	O	Name of NV pair is the BACKED_UP_STATUS of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a boolean BackedUpStatusType of module AlarmIRPConstDefs.
backUpObject	One NV pair of	O	Name of NV pair is the BACK_UP_OBJECT of interface

IS Parameters	OMG CORBA Structured Event attribute	Qualifier	Comment
	filterable_ body_fields		AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a string carrying of DN of the back-up object. See 3G TS 32.300 [3] for the DN string representation.
trendIndication	One NV pair of filterable_ body_fields	O	Name of NV pair is the TREND_INDICATION of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is an enum TrendIndicationType of module AlarmIRPConstDefs.
thresholdInfo	One NV pair of filterable_ body_fields	O	Name of NV pair is the THRESHOLD_INFO of interface ParameterNameValue of module AlarmIRPConstDefs. Value of NV pair is a ThresholdInfoType of module AlarmIRPConstDefs.
stateChange Definition	One NV pair of filterable_ body_fields	O	Name of NV pair is the STATE_CHANGE_DEFINITION of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is an AttributeChangeSetType of module AlarmIRPConstDefs.
monitoredAttributes	One NV pair of filterable_ body_fields	O	Name of NV pair is the MONITORED_ATTRIBUTES of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is an AttributeSetType of module AlarmIRPConstDefs.
proposedRepairActions	One NV pair of filterable_ body_fields	O	Name of NV pair is the PROPOSED_REPAIR_ACTIONS of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a string.
additionalText	One NV pair of filterable_ body_fields	O	Name of NV pair is the ADDITIONAL_TEXT of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a string.
alarmId	One NV pair of filterable_ body_fields	M	Name of NV pair is the ALARM_ID of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a string. If the string is a zero-length string or if this NV pair is absent, the default semantics is that alarmId is a concatenation of managedObjectInstance, eventType, probableCause and specificProblem, if present, of this Structured Event. Since probableCause is encoded as a short, it shall be converted into string before concatenation. The resultant string shall not contain spaces.
There is no corresponding IS attribute.	remaining_ body		

Table 11: Mapping for `notifyAckStateChanged`

IS Parameters	OMG CORBA Structured Event attribute	Qualifier	Comment
There is no corresponding IS attribute.	domain_name		See that of notifyNewAlarm.
notificationType	type_name	M	This is the NOTIFY_FM_ACK_STATE_CHANGED of interface NotificationType of module AlarmIRPConstDefs.
alarmType	event_name	M	See that of notifyNewAlarm.
There is no corresponding IS attribute.	variable Header		
objectClass, objectInstance	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
notification Id	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
eventTime	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
systemDN	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
probableCause	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
perceived Severity	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
alarmId	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
ackTime	One NV pair of filterable_ body_fields	M	Name of NV pair is the ACK_TIME of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a IRPTime of module ManagedGenericIRPConstDefs.
ackUserId	One NV pair of filterable_ body_fields	M	Name of NV pair is the ACK_USER_ID of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a string.
ackSystemId	One NV pair of filterable_ body_fields	O	Name of NV pair is the ACK_SYSTEM_ID of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a string.
ackState	One NV pair of filterable_body_fields	M	Name of NV pair is the ACK_STATE of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a short defined by interface AckState of module AlarmIRPConstDefs.
There is no corresponding IS attribute.	remaining_ body		

Table 12: Mapping for `notifyClearedAlarm`

IS Parameters	OMG CORBA Structured Event attribute	Qualifier	Comment
There is no corresponding IS attribute.	domain_name		See that of notifyNewAlarm.
notificationType	type_name	M	This is the NOTIFY_FM_CLEARED_ALARM of interface NotificationType of module AlarmIRPConstDefs.
alarmType	event_name	M	See that of notifyNewAlarm.
There is no corresponding IS attribute.	variable Header		
objectClass, objectInstance	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
notification Id	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
eventTime	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
systemDN	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
probableCause	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
perceivedSeverity	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
correlatedNotifications	--	--	See Note.
alarmId	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
clearUserId	One NV pair of filterable_ body_fields	O	Name of NV pair is the CLEAR_USER_ID of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a string.
clearSystemId	One NV pair of filterable_ body_fields	O	Name of NV pair is the CLEAR_SYSTEM_ID of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a string.
There is no corresponding IS attribute.	remaining_ body		
NOTE: In the CORBA Solution Set the correlatedNotifications is not used. In the CORBA Solution Set, one notifyClearedAlarm notification can only clear a single alarmInformation.			

Table 13: Mapping for `notifyAlarmListRebuilt`

IS Parameters	OMG CORBA Structured Event attribute	Qualifier	Comment
There is no corresponding IS attribute.	domain_name		See that of notifyNewAlarm.
notificationType	type_name	M	This is the NOTIFY_FM_ALARM_LIST_REBUILT of interface NotificationType of module AlarmIRPConstDefs.
There is no corresponding IS attribute.	event_name	M	Carry an empty string.
There is no corresponding IS attribute.	variable Header		
objectClass, objectInstance	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
notification Id	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
eventTime	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
systemDN	One NV pair of filterable_ body_fields	O	See that of notifyNewAlarm.
reason	One NV pair of filterable_ body_fields	M	Name of NV pair is the REASON of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a string.
There is no corresponding IS attribute.	remaining_ body		

Table 14: Mapping for `notifyChangedAlarm`

IS Parameters	OMG CORBA Structured Event attribute	Qualifier	Comment
There is no corresponding IS attribute.	domain_name		See that of notifyNewAlarm.
notificationType	type_name	M	This is the NOTIFY_FM_CHANGED_ALARM of interface NotificationType of module AlarmIRPConstDefs.
alarmType	event_name	M	See that of notifyNewAlarm.
There is no corresponding IS attribute.	variable Header		
objectClass, objectInstance	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
notification Id	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
eventTime	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
systemDN	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
probableCause	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
perceived Severity	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
alarmId	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
There is no corresponding IS attribute.	remaining_ body		

Table 15: Mapping for `notifyComments`

IS Parameters	OMG CORBA Structured Event attribute	Qualifier	Comment
There is no corresponding IS attribute.	domain_name		See that of notifyNewAlarm.
notificationType	type_name	M	This is the NOTIFY_FM_COMMENT_ADDED of interface NotificationType of module AlarmIRPConstDefs.
alarmType	event_name	M	See that of notifyNewAlarm.
There is no corresponding IS attribute.	variable Header		
objectClass, objectInstance	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
notification Id	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
eventTime	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
systemDN	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
probableCause	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
perceived Severity	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
alarmId	One NV pair of filterable_ body_fields	M	See that of notifyNewAlarm.
comments	One NV pair of filterable_ body_fields	M	Name of NV pair is the COMMENTS of interface AttributeNameValue of module AlarmIRPConstDefs. Value of NV pair is a CommentSet of module AlarmIRPConstDefs.
There is no corresponding IS attribute.	remaining_ body		

6 `AlarmIRPNotifications` Interface

OMG CORBA Notification push operation is used to realise the notification of `AlarmIRPNotifications`. All the notifications in this interface are implemented using this `push_structured_event` method.

6.1 Method `push` (M)

```
module CosNotifyComm {

      …
      Interface SequencePushConsumer : NotifyPublish {
            void push_structured_events(

                  in CosNotification::EventBatch notifications)

          raises( CosEventComm::Disconnected);

             …

      }; // SequencePushConsumer

      …

}; // CosNotifyComm
```

NOTE 1: The `push_structured_events` method takes an input parameter of type `EventBatch` as defined in the OMG `CosNotification` module (OMG Notification Service [1]). This data type is the same as a sequence of Structured Events. Upon invocation, this parameter will contain a sequence of Structured Events being delivered to IRPManager by IRPAgent to which it is connected.

NOTE 2: The maximum number of events that will be transmitted within a single invocation of this operation is controlled by IRPAgent wide configuration parameter.

NOTE 3: The amount of time the supplier (IRPAgent) of a sequence of Structured Events will accumulate individual events into the sequence before invoking this operation is controlled by IRPAgent wide configuration parameter as well.

NOTE 4: IRPAgent may push `EventBatch` with only one Structured Event.

Annex A (normative): IDL specification (file name “AlarmIRPConstDefs.idl”)

```
#ifndef AlarmIRPConstDefs_idl
#define AlarmIRPConstDefs_idl

#include "CosNotification.idl"
#include "ManagedGenericIRPConstDefs.idl"

// This statement must appear after all include statements
#pragma prefix "3gppsa5.org"

/* ## Module: AlarmIRPConstDefs
This module contains commonly used definitions for Alarm IRP
=================================================================
*/
module AlarmIRPConstDefs
{
   /*
   Define the this Alarm IRP version.
   This string is used for the return value of
         get_alarm_IRP_versions().
   It is used as return value of get_notification_categories()
         if the Notification IRP supports the emission of notifications
         defined by this Alarm IRP version.
   It is also used in the domain_name attribute of a structured event
         carrying alarm information defined by this Alarm IRP version.

   See definition "IRP document version number string".
   */
   const string ALARM_IRP_VERSION = "<to be updated using the rule>";

   /*
   This block identifies the alarm types specified for this IRP version.
   These types carry the same semantics as the TMN ITU-T defined event
   types of the same name.
   Their encodings for this version of Alarm IRP are defined here.  Other IRP
   documents, or other versions of Alarm IRP, shall identify their own
   alarm types for their use.  They shall define their encodings
   as well.  Values defined here are unique among themselves.
   */
   interface AlarmType
   {
      const string COMMUNICATIONS_ALARM = "x1";
      const string PROCESSING_ERROR_ALARM = "x2";
      const string ENVIRONMENTAL_ALARM = "x3";
      const string QUALITY_OF_SERVICE_ALARM = "x4";
      const string EQUIPMENT_ALARM = "x5";
   };

   /*
   This block identifies the notification types defined by this
   Alarm IRP version.
   */
   interface NotificationType
   {
      const string NOTIFY_FM_NEW_ALARM = "x1";
      const string NOTIFY_FM_CHANGED_ALARM = "x2";
      const string NOTIFY_FM_ACK_STATE_CHANGED = "x3";
      const string NOTIFY_FM_COMMENT_ADDED = "x4";
```

```
   const string NOTIFY_FM_CLEARED_ALARM = "x5";
   const string NOTIFY_FM_ALARM_LIST_REBUILT = "x6";
};

/*
This block identifies the levels of severity.
*/
interface PerceivedSeverity
{
   const short INDETERMINATE = 1;
   const short CRITICAL = 2;
   const short MAJOR = 3;
   const short MINOR = 4;
   const short WARNING = 5;
   const short CLEARED = 6;
};

/*
This block identifies the probable cause of a reported alarm.
*/
interface ProbableCause
{
   const short ALARM_INDICATION_SIGNAL = 1;
   const short CALL_SETUP_FAILURE = 2;
   const short DEGRADED_SIGNAL_M3100 = 3;
   const short FAR_END_RECEIVER_FAILURE = 4;
   const short FRAMING_ERROR_M3100 = 5;
   const short LOSS_OF_FRAME = 6;
   const short LOSS_OF_POINTER = 7;
   const short LOSS_OF_SIGNAL = 8;
   const short PAYLOAD_TYPE_MISMATCH = 9;
   const short TRANSMISSION_ERROR = 10;
   const short REMOTE_ALARM_INTERFACE = 11;
   const short EXCESSIVE_BIT_ERROR_RATE = 12;
   const short PATH_TRACE_MISMATCH = 13;
   const short UNAVAILABLE = 14;
   const short SIGNAL_LABEL_MISMATCH = 15;
   const short LOSS_OF_MULTI_FRAME = 16;
   const short BACK_PLANE_FAILURE = 51;
   const short DATA_SET_PROBLEM = 52;
   const short EQUIPMENT_IDENTIFIER_DUPLICATION = 53;
   const short EXTERNAL_DEVICE_PROBLEM = 54;
   const short LINE_CARD_PROBLEM = 55;
   const short MULTIPLEXER_PROBLEM_M3100 = 56;
   const short NE_IDENTIFIER_DUPLICATION = 57;
   const short POWER_PROBLEM_M3100 = 58;
   const short PROCESSOR_PROBLEM_M3100 = 59;
   const short PROTECTION_PATH_FAILURE = 60;
   const short RECEIVER_FAILURE_M3100 = 61;
   const short REPLACEABLE_UNIT_MISSING = 62;
   const short REPLACEABLE_UNIT_TYPE_MISMATCH = 63;
   const short SYNCHRONISATION_SOURCE_MISMATCH = 64;
   const short TERMINAL_PROBLEM = 65;
   const short TIMING_PROBLEM_M3100 = 66;
   const short TRANSMITTER_FAILURE_M3100 = 67;
   const short TRUNK_CARD_PROBLEM = 68;
   const short REPLACEABLE_UNIT_PROBLEM = 69;
   const short AIR_COMPRESSOR_FAILURE = 101;
   const short AIR_CONDITIONING_FAILURE = 102;
   const short AIR_DRYER_FAILURE = 103;
   const short BATTERY_DISCHARGING = 104;
   const short BATTERY_FAILURE = 105;
   const short COMMERICAL_POWER_FAILURE = 106;
```

```
const short COOLING_FAN_FAILURE = 107;
const short ENGINE_FAILURE = 108;
const short FIRE_DETECTOR_FAILURE = 109;
const short FUSE_FAILURE = 110;
const short GENERATOR_FAILURE = 111;
const short LOW_BATTERY_THRESHOLD = 112;
const short PUMP_FAILURE_M3100 = 113;
const short RECTIFIER_FAILURE = 114;
const short RECTIFIER_HIGH_VOLTAGE = 115;
const short RECTIFIER_LOW_F_VOLTAGE = 116;
const short VENTILATION_SYSTEM_FAILURE = 117;
const short ENCLOSURE_DOOR_OPEN_M3100 = 118;
const short EXPLOSIVE_GAS = 119;
const short FIRE = 120;
const short FLOOD = 121;
const short HIGH_HUMIDITY = 122;
const short HIGH_TEMPERATURE = 123;
const short HIGH_WIND = 124;
const short ICE_BUILD_UP = 125;
const short INTRUSION_DETECTION = 126;
const short LOW_FUEL = 127;
const short LOW_HUMIDITY = 128;
const short LOW_CABLE_PRESSURE = 129;
const short LOW_TEMPERATURE = 130;
const short LOW_WATER = 131;
const short SMOKE = 132;
const short TOXIC_GAS = 133;
const short STORAGE_CAPACITY_PROBLEM_M3100 = 151;
const short MEMORY_MISMATCH = 152;
const short CORRUPT_DATA_M3100 = 153;
const short OUT_OF_CPU_CYCLES = 154;
const short SOFTWARE_ENVIRONMENT_PROBLEM = 155;
const short SOFTWARE_DOWNLOAD_FAILURE = 156;
const short ADAPTER_ERROR = 301;
const short APPLICATION_SUBSYSTEM_FAILURE = 302;
const short BANDWIDTH_REDUCTION = 303;
const short COMMUNICATION_PROTOCOL_ERROR = 305;
const short COMMUNICATION_SUBSYSTEM_FAILURE = 306;
const short CONFIGURATION_OR_CUSTOMIZING_ERROR = 307;
const short CONGESTION = 308;
const short CPU_CYCLES_LIMIT_EXCEEDED = 310;
const short DATA_SET_OR_MODEM_ERROR = 311;
const short DTE_DCE_INTERFACE_ERROR = 313;
const short EQUIPMENT_MALFUNCTION = 315;
const short EXCESSIVE_VIBRATION = 316;
const short FILE_ERROR = 317;
const short HEATING_OR_VENTILATION_OR_COOLING_SYSTEM_PROBLEM = 321;
const short HUMIDITY_UNACCEPTABLE = 322;
const short INPUT_OUTPUT_DEVICE_ERROR = 323;
const short INPUT_DEVICE_ERROR = 324;
const short LAN_ERROR = 325;
const short LEAK_DETECTION = 326;
const short LOCAL_NODE_TRANSMISSION_ERROR = 327;
const short MATERIAL_SUPPLY_EXHAUSTED = 330;
const short OUT_OF_MEMORY = 332;
const short OUTPUT_DEVICE_ERROR = 333;
const short PERFORMANCE_DEGRADED = 334;
const short PRESSURE_UNACCEPTABLE = 336;
const short QUEUE_SIZE_EXCEEDED = 339;
const short RECEIVE_FAILURE = 340;
const short REMOTE_NODE_TRANSMISSION_ERROR = 342;
const short RESOURCE_AT_OR_NEARING_CAPACITY = 343;
const short RESPONSE_TIME_EXCESSIVE = 344;
```

```
const short RETRANSMISSION_RATE_EXCESSIVE = 345;
const short SOFTWARE_ERROR = 346;
const short SOFTWARE_PROGRAM_ABNORMALLY_TERMINATED = 347;
const short SOFTWARE_PROGRAM_ERROR = 348;
const short TEMPERATURE_UNACCEPTABLE = 350;
const short THRESHOLD_CROSSED = 351;
const short TOXIC_LEAK_DETECTED = 353;
const short TRANSMIT_FAILURE = 354;
const short UNDERLYING_RESOURCE_UNAVAILABLE = 356;
const short VERSION_MISMATCH = 357;
const short A_BIS_TO_BTS_INTERFACE_FAILURE = 501;
const short A_BIS_TO_TRX_INTERFACE_FAILURE = 502;
const short ANTENNA_PROBLEM = 503;
const short BATTERY_BREAKDOWN = 504;
const short BATTERY_CHARGING_FAULT = 505;
const short CLOCK_SYNCHRONISATION_PROBLEM = 506;
const short COMBINER_PROBLEM = 507;
const short DISK_PROBLEM = 508;
const short EXCESSIVE_RECEIVER_TEMPERATURE = 510;
const short EXCESSIVE_TRANSMITTER_OUTPUT_POWER = 511;
const short EXCESSIVE_TRANSMITTER_TEMPERATURE = 512;
const short FREQUENCY_HOPPING_DEGRADED = 513;
const short FREQUENCY_HOPPING_FAILURE = 514;
const short FREQUENCY_REDEFINITION_FAILED = 515;
const short LINE_INTERFACE_FAILURE = 516;
const short LINK_FAILURE = 517;
const short LOSS_OF_SYNCHRONISATION = 518;
const short LOST_REDUNDANCY = 519;
const short MAINS_BREAKDOWN_WITH_BATTERY_BACKUP = 520;
const short MAINS_BREAKDOWN_WITHOUT_BATTERY_BACKUP = 521;
const short POWER_SUPPLY_FAILURE = 522;
const short RECEIVER_ANTENNA_FAULT = 523;
const short RECEIVER_MULTICOUPLER_FAILURE = 525;
const short REDUCED_TRANSMITTER_OUTPUT_POWER = 526;
const short SIGNAL_QUALITY_EVALUATION_FAULT = 527;
const short TIMESLOT_HARDWARE_FAILURE = 528;
const short TRANSCEIVER_PROBLEM = 529;
const short TRANSCODER_PROBLEM = 530;
const short TRANSCODER_OR_RATE_ADAPTER_PROBLEM = 531;
const short TRANSMITTER_ANTENNA_FAILURE = 532;
const short TRANSMITTER_ANTENNA_NOT_ADJUSTED = 533;
const short TRANSMITTER_LOW_VOLTAGE_OR_CURRENT = 535;
const short TRANSMITTER_OFF_FREQUENCY = 536;
const short DATABASE_INCONSISTENCY = 537;
const short FILE_SYSTEM_CALL_UNSUCCESSFUL = 538;
const short INPUT_PARAMETER_OUT_OF_RANGE = 539;
const short INVALID_PARAMETER = 540;
const short INVALID_POINTER = 541;
const short MESSAGE_NOT_EXPECTED = 542;
const short MESSAGE_NOT_INITIALISED = 543;
const short MESSAGE_OUT_OF_SEQUENCE = 544;
const short SYSTEM_CALL_UNSUCCESSFUL = 545;
const short TIMEOUT_EXPIRED = 546;
const short VARIABLE_OUT_OF_RANGE = 547;
const short WATCH_DOG_TIMER_EXPIRED = 548;
const short COOLING_SYSTEM_FAILURE = 549;
const short EXTERNAL_EQUIPMENT_FAILURE = 550;
const short EXTERNAL_POWER_SUPPLY_FAILURE = 551;
const short EXTERNAL_TRANSMISSION_DEVICE_FAILURE = 552;
const short REDUCED_ALARM_REPORTING = 561;
const short REDUCED_EVENT_REPORTING = 562;
const short RECUCED_LOGGING_CAPABILITY = 563;
const short SYSTEM_RESOURCES_OVERLOAD = 564;
```

```
   const short BROADCAST_CHANNEL_FAILURE = 565;
   const short CALL_ESTABLISHMENT_ERROR = 566;
   const short INVALID_MESSAGE_RECEIVED = 567;
   const short INVALID_MSU_RECEIVED = 568;
   const short LAPD_LINK_PROTOCOL_FAILURE = 569;
   const short LOCAL_ALARM_INDICATION = 570;
   const short REMOTE_ALARM_INDICATION = 571;
   const short ROUTING_FAILURE = 572;
   const short SS7_PROTOCOL_FAILURE = 573;
   const short TRANSMISSION_FAILURE = 574;
};

/*
This block identifies the acknowledgement state of a reported alarm.
*/
interface AckState
{
   const short ACKNOWLEDGED = 1;
   const short UNACKNOWLEDGED = 2;
};

/*
This block identifies attributes which are included as part of the Alarm IRP
These attribute values should not clash with those defined for the attributes
of notification header (see IDL of Notification IRP).
*/
interface AttributeNameValue
{
   const string ALARM_ID = "f";
   const string PROBABLE_CAUSE = "g";
   const string PERCEIVED_SEVERITY = "h";
   const string SPECIFIC_PROBLEM = "i";
   const string ADDITIONAL_TEXT = "j";
   const string ACK_TIME = "k";
   const string ACK_USER_ID = "l";
   const string ACK_SYSTEM_ID = "m";
   const string ACK_STATE = "n";
   const string COMMENTS = "o";
   const string BACKED_UP_STATUS = "p";
   const string BACK_UP_OBJECT = "q";
   const string THRESHOLD_INFO = "r";
   const string TREND_INDICATION = "s";
   const string STATE_CHANGE_DEFINITION = "t";
   const string MONITORED_ATTRIBUTES = "u";
   const string PROPOSED_REPAIR_ACTIONS = "v";
   const string CORRELATED_NOTIFICATIONS = "w";
   const string REASON = "x";
   const string CLEAR_USER_ID = “Y”;
   const string CLEAR_SYSTEM_ID = “z”;
};

/*
Defines the content of a Comment
*/
struct Comment
{
   ManagedGenericIRPConstDefs::IRPTime comment_time;
   string comment_text;
   string user_id;
   string system_id;
};

/*
```

```
   Defines a set of comments which are placed in the COMMENTS attribute
   of a structured event.
   */
   typedef sequence <Comment> CommentSet;

   /*
   It indicates if an object has a back up.
   True implies backed up.  False implies not backed up.
   */
   typedef boolean BackedUpStatusType;

   /*
   It indicates if the threshold crossed was in the up or down direction.
   */
   enum ThresholdIndicationType {Up, Down};


/* FloatTypeOpt is an optional type.
   If the discriminator is true the value is present.
   Otherwise the value is null.
*/
union FloatTypeOpt switch (boolean)
{
   case TRUE: float value;
};


/* ThresholdLevelIndType describes multi-level
   threshold crossings.
   Up is the only permitted choice for a counter.
   If indication is “up”, low value is optional.

   @member indication: indicates up or down direction
     of crossing.
   @member low: the low observed value.
   @member high: the high observed value.
*/

struct ThresholdLevelIndType
{
    ThresholdIndicationType indication;
    FloatTypeOpt low;
    float high;
};

/* ThresholdLevelIndTypeOpt is an optional type.
   If the discriminator is true the value is present.
   Otherwise, the value is null.
*/

union ThresholdLevelIndTypeOpt switch (boolean)
{
  case TRUE: ThresholdLevelIndType value;
};


/* ThresholdInfoType indicates some guage or counter
   attribute passed a set threshold.

   @member attributeID: identifies the attribute that
     crossed the threshold.
   @member observedValue: attributes that are of type
     integer will be converted to floats.
```

```
   @member thresholdlevel: This parameter is for
     multi-level threhsolds. Optional.
   @member armTime: May contain empty string.
*/

struct ThresholdInfoType
{
  string attributeID;
  float observedValue;
  ThresholdLevelIndTypeOpt thresholdLevel;
  string armTime;
};


   /*
   It indicates if some observed condition is getting better, worse,
   or not changing.
   */
   enum TrendIndicationType {LessSevere, NoChange, MoreSevere};

   /*
   It is used to report a changed attribute value.
   */
   struct AttributeValueChangeType
   {
      string attribute_name;
      any    old_value;  // type depends on attribute
      any    new_value;  // type depends on attribute
   };

   typedef sequence <AttributeValueChangeType> AttributeChangeSetType;

   /*
   It is used to report an attribute and its value.
   */
   struct AttributeValueType
   {
      string attribute_name;
      any    value;  // type depends on the attribute
   };

   typedef sequence <AttributeValueType> AttributeSetType;

   typedef sequence <long> NotifIdSetType;

   /*
   This holds identifiers of notifications that are correlated.
   */
   struct CorelatedNotification
   {
      string source;  // Contains DN of MO that emitted the set of notifications
                      // DN string format in compliance with Name Convention for
                      // Managed Object.
                      // This may be a zero-length string.  In this case, the MO
                      // is identified by the value of the MOI attribute
                      // of the Structured Event, i.e., the notification.
      NotifIdSetType notif_id_set;  // Set of related notification ids
   };

   /*
   Correlated Notification sets are sets of Correlated Notification
   structures.
   */
```

```
   typedef sequence <CorelatedNotification> CorrelatedNotificationSetType;

   /*
   Define the structure of Alarm ID and Perceived Severity used within the
   alarm acknowledgment operation. Note: perceivedSeverity is an optional
   parameter.
   */
   struct AlarmInformationIdAndSev
   {
      string alarm_information_reference;
      PerceivedSeverity perceived_severity;
   };

   /*
   Define set of the above structure of Alarm ID and Perceived Severity.
   */
   typedef sequence <AlarmInformationIdAndSev> AlarmInformationIdAndSevSeq;

   /*
   It indicates the reason for an alarm acknowledgement to have failed:
     - The specified Alarm Information is absent from the Alarm List
     - The Perceived Severity to be acknowledged has changed and/or is different
       within the Alarm List
     - The acknowledgement failed for some other reason
   */
   enum AcknowledgeFailureCategories
   {
      UnknownAlarmId,
      WrongPerceivedSeverity,
      AcknowledgmentFailed
   };

   /*
   Define the structure returned when an operation fails for a set of alarm ids.
   A reason is provided in order to indicate why the operation failed.
   */
   struct BadAlarmInformationId
   {
      string alarm_information_reference;
      string reason;
   };

   /*
   Define the structure returned when the acknowledge operation fails for a set
   of alarm ids.
   A failure category and a reason are provided in order to indicate why the
   operation failed.
   */
   struct BadAcknowledgeAlarmInfo
   {
      string alarm_information_reference;
      AcknowledgeFailureCategories failure_category;
      string reason;
   };

   typedef sequence <BadAlarmInformationId> BadAlarmInformationIdSeq;
   typedef sequence <BadAcknowledgeAlarmInfo> BadAcknowledgeAlarmInfoSeq;
   typedef sequence <string> AlarmInformationIdSeq;
   typedef CosNotification::EventBatch AlarmInformationSeq;
};
#endif
```

IDL specification (file name “AlarmIRPSystem.idl”)

```
#ifndef AlarmIRPSystem_idl
#define AlarmIRPSystem_idl

#include "AlarmIRPConstDefs.idl"
#include "ManagedGenericIRPSystem.idl"

// This statement must appear after all include statements
#pragma prefix "3gppsa5.org"

/* ## Module: AlarmIRPSystem
This module contains the specification of all operations of Alarm IRP Agent.
================================================================
*/
module AlarmIRPSystem
{
   /*
   System fails to complete the operation.  System can provide reason
   to qualify the exception.  The semantics carried in reason
   is outside the scope of this IRP.
   */
   exception GetAlarmIRPVersions { string reason; };
   exception GetAlarmIRPOperationsProfile { string reason; };
   exception GetAlarmIRPNotificationProfile { string reason; };
   exception AcknowledgeAlarms { string reason; };
   exception UnacknowledgeAlarms { string reason; };
   exception CommentAlarms { string reason; };
   exception ClearAlarms { string reason; };
   exception GetAlarmList { string reason; };
   exception GetAlarmCount { string reason; };
   exception NextAlarmInformations { string reason; };

   /*
   The AlarmInformationIterator is used to iterate through a snapshot of
   Alarm Informations taken from the Alarm List when IRPManager invokes
   get_alarm_list. IRPManager uses it to pace the return of Alarm
   Informations.

   IRPAgent controls the life-cycle of the iterator. However, a destroy
   operation is provided to handle the case where IRPManager wants to stop
   the iteration procedure before reaching the last iteration.
   */
   interface AlarmInformationIterator
   {
      /*
      This method returns between 1 and "how_many" Alarm Informations. The
      IRPAgent may return less than "how_many" items even if there are more
      items to return. "how_many" must be non-zero. Return TRUE if there may
      be more Alarm Information to return. Return FALSE if there are no more
      Alarm Information to be returned.

      If FALSE is returned, the IRPAgent will automatically destroy the
      iterator.
      */
      boolean next_alarmInformations (
         in unsigned short how_many,
         out AlarmIRPConstDefs::AlarmInformationSeq alarm_informations
      )
      raises (NextAlarmInformations, ManagedGenericIRPSystem::InvalidParameter);

      /*
      This method destroys the iterator.
```

```
   */
   void destroy();
};


interface AlarmIRP
{
   /*
   Return the list of all supported Alarm IRP versions.
   */
   ManagedGenericIRPConstDefs::VersionNumberSet get_alarm_IRP_versions (
   )
   raises (GetAlarmIRPVersions);


   /*
   Return the list of all supported operations and their supported
   parameters for a specific Alarm IRP version.
   */
   ManagedGenericIRPConstDefs::MethodList get_alarm_IRP_operations_profile (
      in ManagedGenericIRPConstDefs::VersionNumber alarm_irp_version
   )
   raises (GetAlarmIRPOperationsProfile,
           ManagedGenericIRPSystem::OperationNotSupported,
           ManagedGenericIRPSystem::InvalidParameter);

   /*
   Return the list of all supported notifications and their supported
   parameters for a specific Alarm IRP version.
   */
   ManagedGenericIRPConstDefs::MethodList get_alarm_IRP_notification_profile
   (
      in ManagedGenericIRPConstDefs::VersionNumber alarm_irp_version
   )
   raises (GetAlarmIRPNotificationProfile,
           ManagedGenericIRPSystem::OperationNotSupported,
           ManagedGenericIRPSystem::InvalidParameter);


   /*
   Request to acknowledge one or more alarms.
   */
   ManagedGenericIRPConstDefs::Signal acknowledge_alarms (
      in AlarmIRPConstDefs::AlarmInformationIdAndSevSeq
         alarm_information_id_and_sev_list,
      in string ack_user_id,
      in string ack_system_id,
      out AlarmIRPConstDefs::BadAcknowledgeAlarmInfoSeq
         bad_ack_alarm_info_list
   )
   raises (AcknowledgeAlarms, ManagedGenericIRPSystem::ParameterNotSupported,
           ManagedGenericIRPSystem::InvalidParameter);


   /*
   Request to remove acknowledgement information of one or more alarms.
   */
   ManagedGenericIRPConstDefs::Signal unacknowledge_alarms (
      in AlarmIRPConstDefs::AlarmInformationIdSeq alarm_information_id_list,
      in string ack_user_id,
      in string ack_system_id,
      out AlarmIRPConstDefs::BadAlarmInformationIdSeq
          bad_alarm_information_id_list
```

```
)
raises (UnacknowledgeAlarms,
        ManagedGenericIRPSystem::OperationNotSupported,
        ManagedGenericIRPSystem::ParameterNotSupported,
        ManagedGenericIRPSystem::InvalidParameter);


/*
Make comment to one or more alarms.
*/
ManagedGenericIRPConstDefs::Signal comment_alarms (
   in AlarmIRPConstDefs::AlarmInformationIdSeq alarm_information_id_list,
   in string comment_user_id,
   in string comment_system_id,
   in string comment_text,
   out AlarmIRPConstDefs::BadAlarmInformationIdSeq
       bad_alarm_information_id_list
)
raises (CommentAlarms, ManagedGenericIRPSystem::OperationNotSupported,
        ManagedGenericIRPSystem::ParameterNotSupported,
        ManagedGenericIRPSystem::InvalidParameter);

/*
Request to clear one or more alarms.
*/
ManagedGenericIRPConstDefs::Signal clear_alarms (
   in AlarmIRPConstDefs::AlarmInformationIdSeq alarm_information_id_list,
   in string clear_user_id,
   in string clear_system_id,
   out AlarmIRPConstDefs::BadAlarmInformationIdSeq
      bad_alarm_information_id_list
)
raises (ClearAlarms, ManagedGenericIRPSystem::ParameterNotSupported,
        ManagedGenericIRPSystem::InvalidParameter);



/*
This method returns Alarm Informations.
If flag is TRUE, all returned Alarm Informations shall be
in AlarmInformationSeq that contains 0 or more Alarm Informations.
Output parameter iter shall be useless.
If flag is FALSE, no Alarm Informations shall be in AlarmInformationSeq.
IRPAgent needs to use iter to retrieve them.
*/
AlarmIRPConstDefs::AlarmInformationSeq get_alarm_list (
   in string filter,
   out boolean flag,
   out AlarmInformationIterator iter
)
raises (GetAlarmList, ManagedGenericIRPSystem::ParameterNotSupported,
        ManagedGenericIRPSystem::InvalidParameter);


/*
This method returns the count of Alarm Informations.
*/
void get_alarm_count (
   in string filter,
   out unsigned long critical_count,
   out unsigned long major_count,
```

```
         out unsigned long minor_count,
         out unsigned long warning_count,
         out unsigned long indeterminate_count,
         out unsigned long cleared_count
      )
      raises (GetAlarmCount, ManagedGenericIRPSystem::OperationNotSupported,
              ManagedGenericIRPSystem::ParameterNotSupported,
              ManagedGenericIRPSystem::InvalidParameter);
   };
};

#endif
```

Annex B (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
Mar 2000	S_07	SP-000012	--	--	Approved at TSG SA #7 and placed under Change Control	2.0.0	3.0.0
Mar 2000	--	--	--	--	cosmetic	3.0.0	3.0.1
Jun 2000	S_08	SP-000253	005	--	Split of TS - Part 3: Alarm Integration Reference Point (IRP): CORBA Solution Set (SS)	3.0.1	3.1.0
Sep 2000	S_09	SP-000439	003	--	Correct push_structured_event of push_structured_events	3.1.0	3.2.0
Sep 2000	S_09	SP-000439	004	--	Remove the use of interface to encapsulate const strings	3.1.0	3.2.0
Dec 2000	S_10	SP-000521	001	1	Allow “Structured Event Filterable Body Fields” to be absent if parameters are not used	3.2.0	3.3.0
Dec 2000	S_10	SP-000521	002	1	Specific behaviour of the Iterator	3.2.0	3.3.0
Dec 2000	S_10	SP-000521	005	--	Inconsistent qualifiers	3.2.0	3.3.0
Mar 2001	S_11	SP-010032	006	--	Missing how "Notify Alarm List Rebuilt" reason attribute is located in Structured Event	3.3.0	3.4.0
Mar 2001	S_11	SP-010032	007	--	Use alarmInformationBody in additionalInformation.ackTime	3.3.0	3.4.0
Jun 2001	S_12	SP-010239	008	--	Probable Cause “Intrusion Detection” is missing	3.4.0	3.5.0
Jun 2001	S_12	SP-010282	009	--	Alarm IRP: CORBA SS Rel4 - Addition of feature.	3.5.1	4.0.0
Sep 2001	S_13	SP-010469	010	--	Correction of BadAlarmInformationIdSeq parameter type	4.0.0	4.1.0
Sep 2001	S_13	SP-010474	011	--	Definition of thresholdInfo in Alarm IRP: CORBA SS	4.0.0	4.1.0
Sep 2001	S_13	SP-010522	012	--	Eliminate guesses on IDL file names in Alarm IRP: CORBA SS	4.0.0	4.1.0
Mar 2002	S_15	SP-020015	014	--	Correction of erroneous and addition of missing mapping tables	4.1.0	4.2.0
Mar 2002	S_15	SP-020028	015	--	Addition of "perceivedSeverity" as parameter to "acknowledgeAlarms" operation (CORBA SS)	4.1.0	4.2.0
Mar 2002	S_15	--	--	--	Automatic upgrade to Rel-5 (no Rel-5 CR)	4.2.0	5.0.0