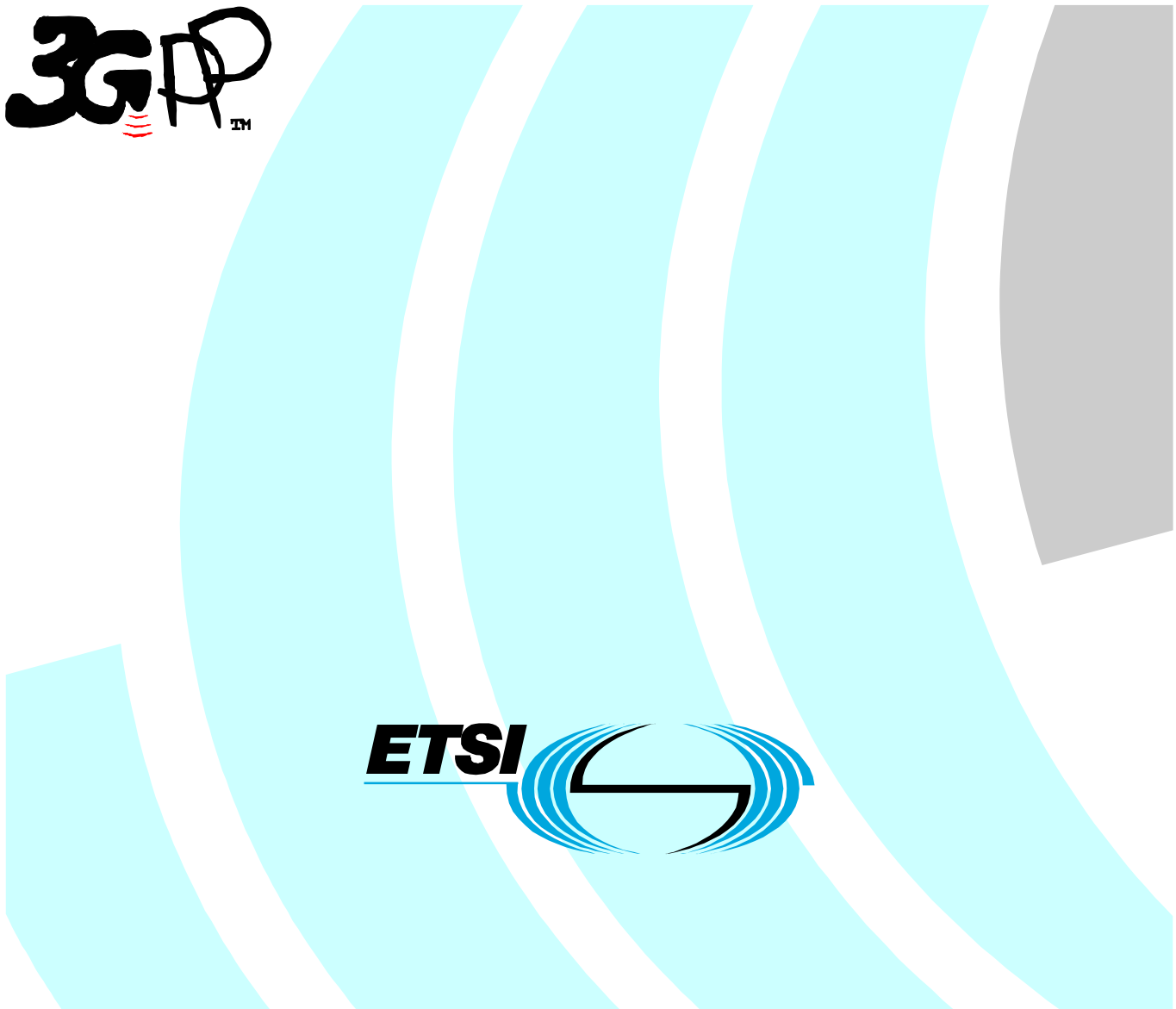
ETSI TS 132 111-3 V4.6.0 (2003-03)

Technical Specification

Universal Mobile Telecommunications System (UMTS);
Telecommunication management;
Fault Management;
Part 3: Alarm Integration Reference Point:
CORBA solution set
(3GPP TS 32.111-3 version 4.6.0 Release 4)



Reference
RTS/TSGS-0532111-3v460

Keywords
UMTS

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

All published ETSI deliverables shall include information which directs the reader to the above source of information.

Foreword

This Technical Specification (TS) has been produced by ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities, UMTS identities or GSM identities. These should be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between GSM, UMTS, 3GPP and ETSI identities can be found under http://webapp.etsi.org/key/queryform.asp .

Contents

Foreword

This Technical Specification (TS) has been produced by the 3rd Generation Partnership Project (3GPP).

The present document is part 3 of a multi-part deliverable covering the 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects, Telecommunication management; Fault management as identified below:

Part 1: "3G Fault management requirements";

Part 2: "Alarm Integration Reference Point: Information Service";

Part 3: "Alarm Integration Reference Point: CORBA Solution Set";

Part 4: "Alarm Integration Reference Point: CMIP Solution Set".

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

x the first digit:

1 presented to TSG for information;

2 presented to TSG for approval;

3 or greater indicates TSG approved document under change control.

y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

z the third digit is incremented when editorial only changes have been incorporated in the document.

1 Scope

The present document specifies the CORBA Solution Set (SS) for the IRP whose semantics is specified in Alarm IRP: Information Service (IS) (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.

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] OMG TC Document telecom/98-11-01: "OMG Notification Service". http://www.omg.org/technology/documents/

[2] OMG CORBA Services: "Common Object Services Specification, Update: November 22, 1996" (Clause 4 contains the Event Service specification). http://www.omg.org/technology/documents/

[3] 3GPP TS 32.300: "Telecommunication management; Configuration Management (CM); Name convention for Managed Objects".

[4] 3GPP TS 32.302: "Telecommunication Management; Configuration Management; Notification Integration Reference Point; Information Service version 1".

[5] 3GPP TS 32.303: "Telecommunication management; Configuration Management (CM); Notification Integration Reference Point; CORBA solution set version 1:1".

[6] 3GPP TS 32.111-2: "Telecommunication management; Fault Management; Part 2: Alarm Integration Reference Point: Information Service".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TS 32.111-2 [6] apply.

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 the present document. The string is derived using the following rule.

Take the 3GPP document number on the front page of the present document, 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 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 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 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 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 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 TS 32.111-2 [13]	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
getOperationProfile	get_alarm_IRP_operations_profile	O
getNotificationProfile	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

5.2 Operation parameter mapping

Reference TS 32.111-2 [6] defines semantics of parameters carried in operations across the Alarm IRP. The following set of tables indicate 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

5.3 Notification parameter mapping

Reference 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.
notificationId	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.

IS Parameters	OMG CORBA Structured Event attribute	Qualifier	Comment
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 filterable_ body_fields	O	Name of NV pair is the BACK_UP_OBJECT of interface 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.
stateChangeDefinition	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.
additionalInformation	One or more NV pairs of filterable_ body_fields	O	Name and value of all NV pairs are vendor-specific.
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

IS Parameters	OMG CORBA Structured Event attribute	Qualifier	Comment
			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.
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 specifications

A.1 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
{

   /*
   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 INDETERMINATE = 0;
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 COMMUNICATIONS_RECEIVE_FAILURE = 17;
const short COMMUNICATIONS_TRANSMIT_FAILURE = 18;
const short MODULATION_FAILURE = 19;
const short DEMODULATION_FAILURE = 20;
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 REAL_TIME_CLOCK_FAILURE = 70;
const short PROTECTION_MECHANISM_FAILURE = 81;
const short PROTECTING_RESOURCE_FAILURE = 82;
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 EXTERNAL_POINT_FAILURE = 136;
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 LOSS_OF_REAL_TIME = 157;
const short REINITIALIZED = 158;
const short EXCESSIVE_ERROR_RATE = 203;
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";
};

/*
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;

   /*
   ShortTypeOpt is a type carrying an optional parameter.
   If the boolean is TRUE, then the value is present.
   Otherwise the value is absent.
   */
   union ShortTypeOpt switch (boolean)
   {
      case TRUE: short value;
   };

   /*
   Define the structure of Alarm ID and Perceived Severity used within the
   alarm acknowledgment operation. Note: perceived_severity is an optional
   parameter. If this value is present, it must have one of the defined values
   of Interface PerceivedSeverity.
   */
   struct AlarmInformationIdAndSev
   {
      string alarm_information_reference;
      ShortTypeOpt 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
```

A.2 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 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.
      Implementations are to provide a return value consisting of one or more IRPVersions.
      Each IRPVersion is defined by the rule in the clause titled
      "IRP document version number string"
      */
```

```
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);

/*
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
Jun 2002	S_16	SP-020282	016	--	Addition of 'indeterminate' probable cause in IDL definition	4.2.0	4.3.0
Sep 2002	S_17	SP-020475	020	--	Correction of CORBA type definition in struct "AlarmInformationIdAndSev"	4.3.0	4.4.0
Dec 2002	S_18	SP-020751	022	--	Add additionalInformation parameter in notification in Alarm IRP: CORBA SS (Alignment with Information Service in Rel-4 32111-2)	4.4.0	4.5.0
Mar 2003	S_19	SP-030064	025	--	Correction of CORBA ALARM_IRP_VERSION in line with adopted Rel-5 policy	4.5.0	4.6.0
Mar 2003	S_19	SP-030062	027	--	Add missing ITU-T M.3100 Probable Cause Values	4.5.0	4.6.0

History

Document history		
V4.0.0	June 2001	Publication
V4.1.0	September 2001	Publication
V4.2.0	March 2002	Publication
V4.3.0	June 2002	Publication
V4.4.0	September 2002	Publication
V4.5.0	December 2002	Publication
V4.6.0	March 2003	Publication