# ETSI TS 132 111-3 V7.0.1 (2007-03)

*Technical Specification*

**Digital cellular telecommunications system (Phase 2+);
Universal Mobile Telecommunications System (UMTS);
Telecommunication management;
Fault Management;
Part 3: Alarm Integration Reference Point (IRP):
Common Object Request Broker Architecture (CORBA)
Solution Set (SS)
(3GPP TS 32.111-3 version 7.0.1 Release 7)**

**GLOBAL SYSTEM FOR
MOBILE COMMUNICATIONS**

Reference
RTS/TSGS-0532111-3v701

Keywords
GSM, 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

*Important notice*

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or
perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF).
In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive
within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status.
Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, please send your comment to one of the following services:
http://portal.etsi.org/chaircor/ETSI_support.asp

*Copyright Notification*

# 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).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

# 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 3<sup>rd</sup> Generation Partnership Project (3GPP).

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.

# Introduction

The present document is part of a TS-family covering the 3<sup>rd</sup> Generation Partnership Project: Technical Specification Group Services and System Aspects; Telecommunication management; as identified below:

32.111-1     "Fault Management; Part 1: 3G fault management requirements".

32.111-2     "Fault Management; Part 2: Alarm Integration Reference Point (IRP): Information Service (IS)".

**32.111-3     "Fault Management; Part 3: Alarm Integration Reference Point (IRP): Common Object Request Broker Architecture (CORBA) Solution Set (SS)".**

32.111-5     "Fault Management; Part 5: Alarm Integration Reference Point (IRP): eXtensible Markup Language (XML) definitions".

# 1 Scope

The present contains the CORBA Solution Set for the IRP whose semantics is specified in Alarm IRP: Information Service (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 TS 32.111-2 V7.0.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. 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 (CM); Notification Integration Reference Point (IRP): Information Service (IS)".

[5]       3GPP TS 32.303: " Telecommunication management; Configuration Management (CM); Notification Integration Reference Point (IRP): Common Object Request Broker Architecture (CORBA) Solution Set (SS)".

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

[7]       3GPP TS 32.312: "Telecommunication management; Generic Integration Reference Point (IRP) management; Information Service (IS)".

[8]       3GPP TS 32.311: "Telecommunication management; Generic Integration Reference Point (IRP) management; Requirements".

# 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 Modelling 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 definition of "IRP document version number string" in 3GPP TS 32.311 [8] provides the rule to derive such a string.

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.

# 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 [6] | SS Method | Qualifier |
|---|---|---|
| acknowledgeAlarms | acknowledge_alarms | M |
| unacknowledgeAlarms | unacknowledge_alarms | O |
| getAlarmList | get_alarm_list | M |
| getIRPVersion (note) | 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<br>Note that OMG Notification Service OMG Notification Service [1] defines this method.<br>See clause 6.1 | M |
| notifyClearedAlarm | push_structured_event<br>See clause 6.1 | M |
| notifyChangedAlarm | push_structured_event<br>See clause 6.1 | M |
| notifyAckStateChanged | push_structured_event<br>See clause 6.1 | M |
| notifyAlarmListRebuilt | push_structured_event<br>See clause 6.1 | M |
| notifyComments | push_structured_event<br>See clause 6.1 | O |
| notifyPotentialFaultyAlarmList | push_structured_event<br>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 |
|---|---|---|
| alarmInformationAndSeverityReferenceList | AlarmIRPConstDefs::AlarmInformationIdAndSevSeq alarm_information_id_and_sev_list<br>Note: perceivedSeverity is optional<br>{ alarmId - Mandatory;<br>  perceivedSeverity - Optional<br>} | M |
| ackUserId | string ack_user_id | M |
| ackSystemId | ManagedGenericIRPConstDefs::StringOpt ack_system_id | O |
| badAlarmInformationReferenceList | AlarmIRPConstDefs::BadAcknowledgeAlarmInfoSeq bad_ack_alarm_info_list | M |
| status | ManagedGenericIRPConstDefs::Signal<br>Exceptions:<br>AcknowledgeAlarms,<br>ManagedGenericIRPSystem::ParameterNotSupported,<br>ManagedGenericIRPSystem::InvalidParameter | M |

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

| IS Operation parameter | SS Method parameter | Qualifier |
|---|---|---|
| alarmInformationReferenceList | AlarmIRPConstDefs::AlarmInformationIdSeq alarm_information_id_list | M |
| ackUserId | string ack_user_id | M |
| ackSystemId | ManagedGenericIRPConstDefs::StringOpt ack_system_id | O |
| badAlarmInformationReferenceList | AlarmIRPConstDefs:: BadAlarmInformationIdSeq bad_alarm_information_id_list | M |
| status | ManagedGenericIRPConstDefs::Signal<br>Exceptions:<br>UnacknowledgeAlarms,<br>ManagedGenericIRPSystem::OperationNotSupported,<br>ManagedGenericIRPSystem::ParameterNotSupported,<br>ManagedGenericIRPSystem::InvalidParameter | M |

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

| IS Operation parameter | SS Method parameter | Qualifier |
|---|---|---|
| alarmAckState, filter | ManagedGenericIRPConstDefs::StringOpt filter | O |
| baseObjectClass, baseObjectInstance | AlarmIRPConstDefs::DNOpt base_object | O (Note 1 |
| alarmInformation List | Return value of type AlarmIRPConstDefs::AlarmInformationSeq (Note 2 | M |
| status | Exceptions:<br>GetAlarmList,<br>FilterComplexityLimit,<br>ManagedGenericIRPSystem::ParameterNotSupported,<br>ManagedGenericIRPSystem::InvalidParameter | M |
| Note 1: If notification `notifyAlarmListRebuilt` supports indicating that only a part of the alarm list has been rebuilt then this parameter shall be supported. | | |
| Note 2: Each Structured Event of AlarmIRPConstDefs::AlarmInformationSeq shall have the fields defined in Table 4a or Table 4b. | | |

**Table 4a: Definition of a Structured Event of AlarmIRPConstDefs::AlarmInformationSeq for alarms not related to security**

| IS Parameters | OMG CORBA Structured Event attribute | Qualifier | Comment |
|---|---|---|---|
| There is no corresponding IS attribute. | domain_name | | |
| notificationType | type_name | M | The parameter carries<br>• NOTIFY_FM_NEW_ALARM of interface NotificationType of module AlarmIRPConstDefs in case the alarm has not yet changed and has not yet been cleared.<br>• NOTIFY_FM_CHANGED_ALARM of interface NotificationType of module AlarmIRPConstDefs in case the alarm has changed but has not yet been cleared.<br>• NOTIFY_FM_CLEARED_ALARM of interface NotificationType of module AlarmIRPConstDefs in case the alarm has been cleared but not yet acknowledged. |
| alarmType | See Table 11 | M | See Table 11 |
| There is no corresponding SS attribute. | variable Header | | |
| objectClass, objectInstance | See Table 11 | M | See Table 11 |
| notificationId | See Table 11 | M | See Table 11 |
| eventTime | See Table 11 | M | Value of NV pair is a IRPTime of module ManagedGenericIRPConstDefs.<br><br>The parameter carries the<br>• alarmRaisedTime in case notificationType carries NOTIFY_FM_NEW_ALARM<br>• alarmChangedTime in case notificationType carries NOTIFY_FM_CHANGED_ALARM<br>• alarmClearedTime in case notificationType carries NOTIFY_FM_CLEARED_ALARM |
| systemDN | See Table 11 | M | See Table 11 |
| alarmId | See Table 11 | M | See Table 11 |
| alarmRaisedTime | One NV pair of filterable_body_fields | M | Name of NV pair is the ALARM_RAISED_TIME of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a IRPTime of module ManagedGenericIRPConstDefs. |
| alarmClearedTime | One NV pair of filterable_body_fields | M | present if related alarm was cleared; not present if related alarm was not cleared<br><br>Name of NV pair is the ALARM_CLEARED_TIME of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a IRPTime of module ManagedGenericIRPConstDefs. |
| probableCause | See Table 11 | M | See Table 11 |
| perceivedSeverity | See Table 11 | M | See Table 11 |
| specificProblem | See Table 11 | O | See Table 11 |
| backedUpStatus | See Table 11 | O | See Table 11 |
| trendIndication | See Table 11 | O | See Table 11 |
| thresholdInfo | See Table 11 | O | See Table 11 |
| stateChangeDefinition | See Table 11 | O | See Table 11 |
| monitoredAttributes | See Table 11 | O | See Table 11 |
| proposedRepairActions | See Table 11 | O | See Table 11 |
| additionalText | See Table 11 | O | See Table 11 |

| IS Parameters | OMG CORBA Structured Event attribute | Qualifier | Comment |
|---|---|---|---|
| additionalInformation | See Table 11 | O | See Table 11 |
| ackTime | See Table 13 | M | present if related alarm was acknowledged or unacknowledged; not present if related alarm was not acknowledged or unacknowledged<br><br>See Table 13 |
| ackUserId | See Table 13 | M | present if related alarm was acknowledged or unacknowledged; not present if related alarm was not acknowledged or unacknowledged<br><br>See Table 13 |
| ackSystemId | See Table 13 | O | optionally present if related alarm was acknowledged or unacknowledged; not present if related alarm was not acknowledged or unacknowledged<br><br>See Table 13 |
| ackState | See Table 13 | M | present if related alarm was acknowledged or unacknowledged; not present if related alarm was not acknowledged or unacknowledged<br><br>See Table 13 |
| clearUserId | See Table 14 | O | optionally present if related alarm was cleared; not present if related alarm was not cleared<br><br>See Table 14 |
| clearSystemId | See Table 14 | O | optionally present if related alarm was cleared; not present if related alarm was not cleared<br><br>See Table 14 |
| backUpObject | See Table 11 | O | See Table 11 |
| correlatedNotifications | See Table 11 | O | See Table 11 |
| comments | See Table 17 | M | present if related alarm was commented; not present if related alarm was not commented<br><br>See Table 17 |
| There is no corresponding IS attribute. | remaining_ body | | |

**Table 4b: Definition of a Structured Event of AlarmIRPConstDefs::AlarmInformationSeq for alarms related to security**

| IS Parameters | OMG CORBA Structured Event attribute | Qualifier | Comment |
|---|---|---|---|
| There is no corresponding IS attribute. | domain_name | | |
| notificationType | type_name | M | See Table 11<br><br>The parameter carries<br>• NOTIFY_FM_NEW_ALARM of interface NotificationType of module AlarmIRPConstDefs in case the alarm has not yet changed and has not yet been cleared.<br>• NOTIFY_FM_CHANGED_ALARM of interface NotificationType of module AlarmIRPConstDefs in case the alarm has changed but has not yet been cleared.<br>• NOTIFY_FM_CLEARED_ALARM of interface NotificationType of module AlarmIRPConstDefs in case the alarm has been cleared but not yet acknowledged. |
| alarmType | See Table 11 | M | See Table 11 |
| There is no corresponding SS attribute. | variable Header | | |
| objectClass, objectInstance | See Table 11 | M | See Table 11 |
| notificationId | See Table 11 | M | See Table 11 |
| eventTime | See Table 11 | M | See Table 11<br><br>The parameter carries the<br>• alarmRaisedTime in case notificationType carries NOTIFY_FM_NEW_ALARM<br>• alarmChangedTime in case notificationType carries NOTIFY_FM_CHANGED_ALARM<br>• alarmClearedTime in case notificationType carries NOTIFY_FM_CLEARED_ALARM |
| systemDN | See Table 11 | M | See Table 11 |
| alarmId | See Table 11 | M | See Table 11 |
| alarmRaisedTime | One NV pair of filterable_ body_fields | M | Name of NV pair is the ALARM_RAISED_TIME of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a IRPTime of module ManagedGenericIRPConstDefs. |
| alarmClearedTime | One NV pair of filterable_ body_fields | M | present if related alarm was cleared; not present if related alarm was not cleared<br><br>Name of NV pair is the ALARM_CLEARED_TIME of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a IRPTime of module ManagedGenericIRPConstDefs. |
| probableCause | See Table 11 | M | See Table 11 |
| perceivedSeverity | See Table 11 | M | See Table 11 |
| specificProblem | See Table 11 | O | See Table 11 |
| additionalText | See Table 11 | O | See Table 11 |
| additionalInformation | See Table 11 | O | See Table 11 |
| ackTime | See Table 13 | M | present if related alarm was acknowledged or unacknowledged; not present if related alarm was not acknowledged or unacknowledged |

| IS Parameters | OMG CORBA Structured Event attribute | Qualifier | Comment |
|---|---|---|---|
| | | | See Table 13 |
| ackUserId | See Table 13 | M | present if related alarm was acknowledged or unacknowledged; not present if related alarm was not acknowledged or unacknowledged<br><br>See Table 13 |
| ackSystemId | See Table 13 | O | optionally present if related alarm was acknowledged or unacknowledged; not present if related alarm was not acknowledged or unacknowledged<br><br>See Table 13 |
| ackState | See Table 13 | M | present if related alarm was acknowledged or unacknowledged; not present if related alarm was not acknowledged or unacknowledged<br><br>See Table 13 |
| clearUserId | See Table 14 | O | optionally present if related alarm was cleared; not present if related alarm was not cleared<br><br>See Table 14 |
| clearSystemId | See Table 14 | O | optionally present if related alarm was cleared; not present if related alarm was not cleared<br><br>See Table 14 |
| correlatedNotifications | See Table 11 | O | See Table 11 |
| comments | See Table 17 | M | present if related alarm was commented; not present if related alarm was not commented<br><br>See Table 17 |
| serviceUser | See Table 12 | M | See Table 12 |
| serviceProvider | See Table 12 | M | See Table 12 |
| securityAlarmDetector | See Table 12 | M | See Table 12 |
| There is no corresponding IS attribute. | remaining_ body | | |

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

| IS Operation parameter | SS Method parameter | Qualifier |
|---|---|---|
| alarmAckState, filter | ManagedGenericIRPConstDefs::StringOpt 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:<br>GetAlarmCount,<br>FilterComplexityLimit,<br>ManagedGenericIRPSystem::OperationNotSupported,<br>ManagedGenericIRPSystem::ParameterNotSupported,<br>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:<br>GetAlarmIRPVersions | M |

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

| IS Operation parameter | SS Method parameter | Qualifier |
|---|---|---|
| alarmInformationReferenceList | AlarmIRPConstDefs::AlarmInformationIdSeq alarm_information_id_list | M |
| commentUserId | string comment_user_id | M |
| commentSystemId | ManagedGenericIRPConstDefs::StringOpt comment_system_id | O |
| commentText | string comment_text | M |
| badAlarmInformationReferenceList | AlarmIRPConstDefs::BadAlarmInformationIdSeq bad_alarm_information_id_list | M |
| status | ManagedGenericIRPConstDefs::Signal<br>Exceptions:<br>CommentAlarms,<br>ManagedGenericIRPSystem::OperationNotSupported,<br>ManagedGenericIRPSystem::ParameterNotSupported,<br>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:<br>GetAlarmIRPOperationsProfile,<br>ManagedGenericIRPSystem::OperationNotSupported,<br>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:<br>GetAlarmIRPNotificationProfile,<br>ManagedGenericIRPSystem::OperationNotSupported,<br>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<br>Exceptions:<br>ClearAlarms,<br>ManagedGenericIRPSystem::OperationNotSupported,<br>ManagedGenericIRPSystem::ParameterNotSupported,<br>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 11: Mapping for** `notifyNewAlarm` (to carry non-security-related alarms)

| 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.<br>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:<br>communications alarm,<br>processing error alarm,<br>environmental alarm,<br>quality of service alarm and<br>equipment alarm.<br><br>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.<br><br>Name of NV pair is the MANAGED_OBJECT_INSTANCE of interface AttributeNameValue of module NotificationIRPConstDefs.<br><br>Value of NV pair is a string. |
| notificationId | One NV pair of remaining_body | M | Name of NV pair is the NOTIFICATION_ID of interface AttributeNameValue of module NotificationIRPConstDefs.<br><br>Value of NV pair is a long. |

| IS Parameters | OMG CORBA Structured Event attribute | Qualifier | Comment |
|---|---|---|---|
| eventTime | One NV pair of filterable_body_fields | M | Name of NV pair is the EVENT_TIME of interface AttributeNameValue of module NotificationIRPConstDefs.<br><br>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.<br><br>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.<br><br>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.<br><br>Value of NV pair is a short defined by interface PerceivedSeverity of module AlarmIRPConstDefs. |
| specificProblem | One NV pair of remaining_body | O | Name of NV pair is the SPECIFIC_PROBLEM of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string. |
| correlatedNotifications | One NV pair of remaining_body | O | Name of NV pair is the CORRELATED_NOTIFICATIONS of interface AttributeNameValue.<br><br>Value of NV pair is a CorrelatedNotificationSet of module AlarmIRPConstDefs. |
| backedUpStatus | One NV pair of remaining_body | O | Name of NV pair is the BACKED_UP_STATUS of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a boolean BackedUpStatus of module AlarmIRPConstDefs. |
| backUpObject | One NV pair of remaining_body | O | Name of NV pair is the BACK_UP_OBJECT of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>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 remaining_body | O | Name of NV pair is the TREND_INDICATION of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is an enum TrendIndication of module AlarmIRPConstDefs. |
| thresholdInfo | One NV pair of remaining_body | O | Name of NV pair is the THRESHOLD_INFO of interface ParameterNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a ThresholdInfo of module AlarmIRPConstDefs. |
| stateChangeDefinition | One NV pair of remaining_body | O | Name of NV pair is the STATE_CHANGE_DEFINITION of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is an AttributeChangeSet of module AlarmIRPConstDefs. |
| monitoredAttributes | One NV pair of remaining_body | O | Name of NV pair is the MONITORED_ATTRIBUTES of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is an AttributeSet of module AlarmIRPConstDefs. |
| proposedRepairActions | One NV pair of remaining_body | O | Name of NV pair is the PROPOSED_REPAIR_ACTIONS of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string. |

| IS Parameters | OMG CORBA Structured Event attribute | Qualifier | Comment |
|---|---|---|---|
| additionalText | One NV pair of remaining_body | O | Name of NV pair is the ADDITIONAL_TEXT of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string. |
| additionalInformation | One or more NV pairs of remaining_body | O | 3GPP defines two Additional Information (AI) NV pairs for carrying  vendor-specific (VS) perceived severity and alarm type.<br>The names of these two NV pairs are:<br>&bull; AI_VS_PERCEIVED_SEVERITY<br>&bull; AI_VS_ALARM_TYPE<br>of interface AdditionalInformation of module  AlarmIRPConstDefs.<br>The values of these two NV pairs are vendor defined.<br>Other AI NV pairs are permitted to capture specific information.<br>To be easily identified as vendor specific, the name value of these NV pairs  should take the form of  "ai_vs_<non-empty string>".<br>The value of this NV pair is vendor defined. |
| alarmId | One NV pair of remaining_body | M | Name of NV pair is the ALARM_ID of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string.<br>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. |

**Table 12: Mapping for `notifyNewAlarm` (to carry security alarm)**

| 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.<br>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:<br>Integrity violation, operational violation, physical violation, security violation and time domain violation.<br><br>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.<br><br>Name of NV pair is the MANAGED_OBJECT_INSTANCE of interface AttributeNameValue of module NotificationIRPConstDefs.<br><br>Value of NV pair is a string. |
| notificationId | One NV pair of remaining_body | M | Name of NV pair is the NOTIFICATION_ID of interface AttributeNameValue of module NotificationIRPConstDefs.<br><br>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.<br><br>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.<br><br>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.<br><br>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.<br><br>Value of NV pair is a short defined by interface PerceivedSeverity of module AlarmIRPConstDefs. |
| correlatedNotifications | One NV pair of remaining_body | O | Name of NV pair is the CORRELATED_NOTIFICATIONS of interface AttributeNameValue.<br><br>Value of NV pair is a CorrelatedNotificationSet of module AlarmIRPConstDefs. |

*ETSI*

| IS Parameters | OMG CORBA Structured Event attribute | Qualifier | Comment |
|---|---|---|---|
| additionalText | One NV pair of remaining_body | O | Name of NV pair is the ADDITIONAL_TEXT of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string. |
| additionalInformation | One or more NV pairs of remaining_body | O | 3GPP defines two Additional Information (AI) NV pairs for carrying vendor-specific (VS) perceived severity and alarm type.<br>The names of these two NV pairs are:<br>    • AI_VS_PERCEIVED_SEVERITY<br>    • AI_VS_ALARM_TYPE<br>of interface AdditionalInformation of module AlarmIRPConstDefs.<br>The values of these two NV pairs are vendor defined.<br>Other AI NV pairs are permitted to capture specific information.<br>To be easily identified as vendor specific, the name value of these NV pairs should take the form of "ai_vs_<non-empty string>".<br>The value of this NV pair is vendor defined. |
| alarmId | One NV pair of remaining_body | M | Name of NV pair is the ALARM_ID of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string.<br>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. |
| serviceUser | One NV pair of remaining_body | M | Name of NV pair is the SERVICE_USER of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string. |
| serviceProvider | One NV pair of remaining_body | M | Name of NV pair is the SERVICE_PROVIDER of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string. |
| securityAlarmDetector | One NV pair of remaining_body | M | Name of NV pair is the SECURITY_ALARM_DETECTOR of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string. |

**Table 13: 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. |
| notificationId | One NV pair of remaining_body | 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. |
| alarmId | One NV pair of remaining_body | M | See that of notifyNewAlarm. |
| ackUserId | One NV pair of remaining_body | M | Name of NV pair is the ACK_USER_ID of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string. |
| ackSystemId | One NV pair of remaining_body | O | Name of NV pair is the ACK_SYSTEM_ID of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string. |
| ackState | One NV pair of remaining_body | M | Name of NV pair is the ACK_STATE of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a short defined by interface AckState of module AlarmIRPConstDefs. |

**Table 14: 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. |
| notificationId | One NV pair of remaining_body | 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 remaining_body | M | See that of notifyNewAlarm. |
| clearUserId | One NV pair of remaining_body | O | Name of NV pair is the CLEAR_USER_ID of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string. |
| clearSystemId | One NV pair of remaining_body | O | Name of NV pair is the CLEAR_SYSTEM_ID of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string. |
| 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. | | | |

*ETSI*

**Table 15: 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. |
| notificationId | One NV pair of remaining_body | 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 remaining_body | M | Name of NV pair is the REASON of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string. |
| alarmListAlignmentRequirement | One NV pair of remaining_body | O | Name of NV pair is the ALARM_LIST_ALIGNMENT_REQUIREMENT of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is an enum AlarmListAlignmentRequirement of module AlarmIRPConstDefs. |

**Table 16: 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. |
| notificationId | One NV pair of remaining_body | 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 remaining_body | M | See that of notifyNewAlarm. |

**Table 17: 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. |
| notificationId | One NV pair of remaining_body | 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. |
| alarmId | One NV pair of remaining_body | M | See that of notifyNewAlarm. |
| comments | One NV pair of remaining_body | M | Name of NV pair is the COMMENTS of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a CommentSet of module AlarmIRPConstDefs. |

**Table 18: Mapping for** `notifyPotentialFaultyAlarmList`

| 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_POTENTIAL_FAULTY_ALARM_LIST of interface NotificationType of module AlarmIRPConstDefs. |
| There is no corresponding IS attribute. | event_name | M | It contains a NULL string. |
| There is no corresponding IS attribute. | variable Header | | |
| objectClass, objectInstance | One NV pair of filterable_body_fields | M | See notifyNewAlarm.<br><br>See sub-clause "Definition" of this notification in [6] for the description of the usage of this field to indicate if part or all AlarmList is potentially faulty. |
| notificationId | One NV pair of remaining_body | M | |
| eventTime | One NV pair of filterable_body_fields | M | See notifyNewAlarm. |
| systemDN | One NV pair of filterable_body_fields | M | See notifyNewAlarm. |
| reason | One NV pair of remaining_body | M | Name of NV pair is the REASON of interface AttributeNameValue of module AlarmIRPConstDefs.<br><br>Value of NV pair is a string. |

# 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")

```
//File: AlarmIRPConstDefs.idl
#ifndef _ALARM_IRP_CONST_DEFS_IDL_
#define _ALARM_IRP_CONST_DEFS_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
{
   /*
   The format of Distinguished Name (DN) is specified in 3GPP TS 32.300
   "Name Conventions for Managed Objects".
   */
   typedef string DN;
   /* DNOpt is an optional type.
   If the discriminator is true the value is present.
   Otherwise the value is null.
   */
   union DNOpt switch (boolean)
   {
      case TRUE: DN value;
   };
   /*
   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";
      const string INTEGRITY_VIOLATION = "x6";
      const string OPERATIONAL_VIOLATION = "x7";
      const string PHYSICAL_VIOLATION = "x8";
      const string SECURITY_SERVICE_OR_MECHANISM_VIOLATION = "x9";
      const string TIME_DOMAIN_VIOLATION = "x10";
   };
   /*
   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";
      const string NOTIFY_FM_POTENTIAL_FAULTY_ALARM_LIST = "x7";
   };
   /*
   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
    {
        /*
        Probable causes originating from M.3100.
        Values below correspond to M.3100 values.
        */
        const short INDETERMINATE = 0;
        const short ALARM_INDICATION_SIGNAL = 1;
        const short CALL_SETUP_FAILURE = 2;
        const short DEGRADED_SIGNAL = 3;
        const short FAR_END_RECEIVER_FAILURE = 4;
        const short FRAMING_ERROR = 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;
        // Values 10 correspond to a duplicated probable cause
        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;
        // Values 21-26 correspond to duplicated probable causes
        // Values 27-50 are reserved for M.3100 potential future extensions
        const short BACK_PLANE_FAILURE = 51;
        const short DATA_SET_PROBLEM = 52;
        const short EQUIPMENT_IDENTIFIER_DUPLICATION = 53;
        const short EXTERNAL_IF_DEVICE_PROBLEM = 54;
        const short LINE_CARD_PROBLEM = 55;
        const short MULTIPLEXER_PROBLEM = 56;
        const short NE_IDENTIFIER_DUPLICATION = 57;
        const short POWER_PROBLEM = 58;
        const short PROCESSOR_PROBLEM = 59;
        const short PROTECTION_PATH_FAILURE = 60;
        const short RECEIVER_FAILURE = 61;
        const short REPLACEABLE_UNIT_MISSING = 62;
        const short REPLACEABLE_UNIT_TYPE_MISMATCH = 63;
        const short SYNCHRONIZATION_SOURCE_MISMATCH = 64;
        const short TERMINAL_PROBLEM = 65;
        const short TIMING_PROBLEM = 66;
        const short TRANSMITTER_FAILURE = 67;
        const short TRUNK_CARD_PROBLEM = 68;
        const short REPLACEABLE_UNIT_PROBLEM = 69;
        const short REAL_TIME_CLOCK_FAILURE = 70;
        // Values 71-80 correspond to duplicated probable causes
        const short PROTECTION_MECHANISM_FAILURE = 81;
        const short PROTECTING_RESOURCE_FAILURE = 82;
        // Values 83-100 are reserved for M.3100 potential future extensions
        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 COMMERCIAL_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 = 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 = 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;
        // Values 134-135 correspond to duplicated probable causes
        const short EXTERNAL_POINT_FAILURE = 136;
        // Values 137-150 are reserved for potential M.3100 future extensions
        const short STORAGE_CAPACITY_PROBLEM = 151;
        const short MEMORY_MISMATCH = 152;
        const short CORRUPT_DATA = 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;
        // Values 159-167 correspond to duplicated probable causes
        // Values 168-200 are reserved for potential M.3100 future extensions
        // Values 201-202 correspond to duplicated probable causes
        const short EXCESSIVE_ERROR_RATE = 203;
        // Values 204-207 correspond to duplicated probable causes
        // Values 208-300 are reserved for potential M.3100 future extensions
        /*
        Probable causes originating from X.721.
        Values below correspond to X.721 values with an offset of 300.
        */
        const short ADAPTER_ERROR = 301;
        const short APPLICATION_SUBSYSTEM_FAILURE = 302;
        const short BANDWIDTH_REDUCED = 303;
        // Value 304 corresponds to a duplicated probable cause
        const short COMMUNICATIONS_PROTOCOL_ERROR = 305;
        const short COMMUNICATIONS_SUBSYSTEM_FAILURE = 306;
        const short CONFIGURATION_OR_CUSTOMIZATION_ERROR = 307;
        const short CONGESTION = 308;
        // Value 309 corresponds to a duplicated probable cause
        const short CPU_CYCLES_LIMIT_EXCEEDED = 310;
        const short DATA_SET_OR_MODEM_ERROR = 311;
        // Value 312 corresponds to a duplicated probable cause
        const short DTE_DCE_INTERFACE_ERROR = 313;
        // Value 314 corresponds to a duplicated probable cause
        const short EQUIPMENT_MALFUNCTION = 315;
        const short EXCESSIVE_VIBRATION = 316;
        const short FILE_ERROR = 317;
        // Values 318-320 correspond to duplicated probable causes
        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_DETECTED = 326;
        const short LOCAL_NODE_TRANSMISSION_ERROR = 327;
        // Values 328-329 correspond to duplicated probable causes
        const short MATERIAL_SUPPLY_EXHAUSTED = 330;
        // Value 331 corresponds to a duplicated probable cause
        const short OUT_OF_MEMORY = 332;
        const short OUTPUT_DEVICE_ERROR = 333;
        const short PERFORMANCE_DEGRADED = 334;
        // Value 335 corresponds to a duplicated probable cause
        const short PRESSURE_UNACCEPTABLE = 336;
        // Values 337-338 correspond to duplicated probable causes
        const short QUEUE_SIZE_EXCEEDED = 339;
        const short RECEIVE_FAILURE = 340;
        // Value 341 corresponds to a duplicated probable cause
        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;
// Value 349 corresponds to a duplicated probable cause
const short TEMPERATURE_UNACCEPTABLE = 350;
const short THRESHOLD_CROSSED = 351;
// Value 352 corresponds to a duplicated probable cause
const short TOXIC_LEAK_DETECTED = 353;
const short TRANSMIT_FAILURE = 354;
// Value 355 corresponds to a duplicated probable cause
const short UNDERLYING_RESOURCE_UNAVAILABLE = 356;
const short VERSION_MISMATCH = 357;
// Values 358-500 are reserved for potential X.721 future extensions
/*
Probable causes for 2G & 3G wireless systems.
*/
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_SYNCHRONIZATION_PROBLEM = 506;
const short COMBINER_PROBLEM = 507;
const short DISK_PROBLEM = 508;
// Value 509 corresponds to a duplicated probable cause
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_SYNCHRONIZATION = 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;
// Value 524 corresponds to a duplicated probable cause
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;
// Value 534 corresponds to a duplicated probable cause
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_INITIALIZED = 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;
// Values 553-560 correspond to duplicated probable causes
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 CONNECTION_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_ERROR = 574;
    // Value 575 corresponds to a duplicated probable cause
    // Values 576-700 are reserved for potential future extensions
    // for 2G & 3G wireless systems
    /*
    Probable causes originating from M.3100 security alarm causes.
    Values below correspond to M.3100 values with an offset of 700.
    */
    const short AUTHENTICATION_FAILURE = 701;
    const short BREACH_OF_CONFIDENTIALITY = 702;
    const short CABLE_TAMPER = 703;
    const short DELAYED_INFORMATION = 704;
    const short DENIAL_OF_SERVICE = 705;
    const short DUPLICATE_INFORMATION = 706;
    const short INFORMATION_MISSING = 707;
    const short INFORMATION_MODIFICATION_DETECTED = 708;
    const short INFORMATION_OUT_OF_SEQUENCE = 709;
    // Value 710 corresponds to a duplicated probable cause
    const short KEY_EXPIRED = 711;
    const short NON_REPUDIATION_FAILURE = 712;
    const short OUT_OF_HOURS_ACTIVITY = 713;
    const short OUT_OF_SERVICE = 714;
    const short PROCEDURAL_ERROR = 715;
    const short UNAUTHORISED_ACCESS_ATTEMPT = 716;
    const short UNEXPECTED_INFORMATION = 717;
    const short UNSPECIFIED_REASON = 718;
    // Values 719-800 are reserved for potential M.3100 future extensions
};
/*
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";
    const string ALARM_LIST_ALIGNMENT_REQUIREMENT = "ff";
    const string SERVICE_USER = "gg";
    const string SERVICE_PROVIDER = "hh";
    const string SECURITY_ALARM_DETECTOR = "ii";
          const string ALARM_RAISED_TIME = "kk";
    const string ALARM_CLEARED_TIME = "ll";
};
```

```
/*
   constants for use in populating the additional information
   name field Names
*/
interface AdditionalInformation
 {
    const string AI_VS_PERCEIVED_SEVERITY = "ai_ps";
    const string AI_VS_ALARM_TYPE         = "ai_at";
 };


/*
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 BackedUpStatus;
/*
It indicates if the threshold crossed was in the up or down direction.
*/
enum ThresholdIndication {UP, DOWN};
/*
It indicates if the AlarmList alignment is required.
*/
enum AlarmListAlignmentRequirement {REQUIRED, NOTREQUIRED};
/* FloatOpt is an optional type.
If the discriminator is true the value is present.
Otherwise the value is null.
*/
union FloatOpt switch (boolean)
{
   case TRUE: float value;
};
/* ThresholdLevelInd 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 ThresholdLevelInd
{
   ThresholdIndication indication;
   FloatOpt low;
   float high;
};
/* ThresholdLevelIndOpt is an optional type.
If the discriminator is true the value is present.
Otherwise, the value is null.
*/
union ThresholdLevelIndOpt switch (boolean)
{
   case TRUE: ThresholdLevelInd value;
};
/* ThresholdInfo indicates some gauge or counter
attribute passed a set threshold.
@member attribute_id: identifies the attribute that
  crossed the threshold.
@member observed_value: attributes that are of type
  integer will be converted to floats.
@member threshold_level: This parameter is for
  multi-level thresholds. Optional.
@member arm_time: May contain empty string.
```

```
*/
struct ThresholdInfo
{
   string attribute_id;
   float observed_value;
   ThresholdLevelIndOpt threshold_level;
   string arm_time;
};
/*
It indicates if some observed condition is getting better, worse,
or not changing.
*/
enum TrendIndication {LESS_SEVERE, NO_CHANGE, MORE_SEVERE};
/*
It is used to report a changed attribute value.
*/
struct AttributeValueChange
{
   string attribute_name;
   any old_value;  // type depends on attribute
   any new_value;  // type depends on attribute
};
typedef sequence <AttributeValueChange> AttributeChangeSet;
/*
It is used to report an attribute and its value.
*/
struct AttributeValue
{
   string attribute_name;
   any value;  // type depends on the attribute
};
typedef sequence <AttributeValue> AttributeSet;
typedef sequence <long> NotifIdSet;
/*
This holds identifiers of notifications that are correlated.
*/
struct CorrelatedNotification
{
   DN 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.
   NotifIdSet notif_id_set;  // Set of related notification ids
};
/*
Correlated Notification sets are sets of Correlated Notification
structures.
*/
typedef sequence <CorrelatedNotification> CorrelatedNotificationSet;
/*
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_id;
   ManagedGenericIRPConstDefs::ShortOpt 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
{
   UNKNOWN_ALARM_ID,
   WRONG_PERCEIVED_SEVERITY,
   ACKNOWLEDGMENT_FAILED
```

```
   };
   /*
   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_id;
      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_id;
      AcknowledgeFailureCategories failure_category;
      string reason;
   };
   typedef sequence <BadAlarmInformationId> BadAlarmInformationIdSeq;
   typedef sequence <BadAcknowledgeAlarmInfo> BadAcknowledgeAlarmInfoSeq;
   typedef sequence <string> AlarmInformationIdSeq;
   typedef CosNotification::EventBatch AlarmInformationSeq;
   /*
   Define the cause values of notifyPotentialFaultyAlarmList and
   notifyAlarmListRebuilt, which have been defined in 32111-2.
   */
   const string AGENT_NE_COMMUNICATION_ERROR = "Agent-NE communication error";
   const string AGENT_RESTARTS = "Agent restarts";
   const string INDETERMINATE = "Indeterminate";
};
#endif // _ALARM_IRP_CONST_DEFS_IDL_
```

# A.2    IDL specification (file name "AlarmIRPSystem.idl")

```
//File: AlarmIRPSystem.idl
#ifndef _ALARM_IRP_SYSTEM_IDL_
#define _ALARM_IRP_SYSTEM_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; };
   exception FilterComplexityLimit { 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_alarm_informations (
         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 ManagedGenericIRPConstDefs::StringOpt 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 ManagedGenericIRPConstDefs::StringOpt 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 ManagedGenericIRPConstDefs::StringOpt 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 ManagedGenericIRPConstDefs::StringOpt clear_system_id,
            out AlarmIRPConstDefs::BadAlarmInformationIdSeq
                bad_alarm_information_id_list
        )
        raises (ClearAlarms, 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 ManagedGenericIRPConstDefs::StringOpt filter,
            in AlarmIRPConstDefs::DNOpt base_object,
            out boolean flag,
            out AlarmInformationIterator iter
        )
        raises (GetAlarmList, FilterComplexityLimit, ManagedGenericIRPSystem::ParameterNotSupported,
                ManagedGenericIRPSystem::InvalidParameter);

        /*
        This method returns the count of Alarm Informations.
        */
        void get_alarm_count (
            in ManagedGenericIRPConstDefs::StringOpt 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, FilterComplexityLimit, ManagedGenericIRPSystem::OperationNotSupported,
                ManagedGenericIRPSystem::ParameterNotSupported,
                ManagedGenericIRPSystem::InvalidParameter);
    };

};

#endif // _ALARM_IRP_SYSTEM_IDL_
```

# A.3 IDL specification (file name "AlarmIRPNotifications.idl")

```
//File: AlarmIRPNotifications.idl
#ifndef _ALARM_IRP_NOTIFICATIONS_IDL_
#define _ALARM_IRP_NOTIFICATIONS_IDL_

#include <AlarmIRPConstDefs.idl>
#include <NotificationIRPNotifications.idl>

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

/* ## Module: AlarmIRPNotifications
This module contains notifications for Alarm IRP
==================================================================
*/

module AlarmIRPNotifications
{

    interface NotifyNewAlarm: NotificationIRPNotifications::Notify
    {
        const string EVENT_TYPE = "notifyNewAlarm";

        /**
         * This constant defines the name of the probableCause property.
         * The data type for the value of this property
         * is short.
         */
        const string PROBABLE_CAUSE =
            AlarmIRPConstDefs::AttributeNameValue::PROBABLE_CAUSE;

        /**
         * This constant defines the name of the
         * perceivedSeverity property.
         * The data type for the value of this property
         * is short.
         */
        const string PERCEIVED_SEVERITY =
            AlarmIRPConstDefs::AttributeNameValue::PERCEIVED_SEVERITY;


        /**
         * This constant defines the name of the specificProblem
         * property. The data type for the value of this property
         * is string.
         */
        const string SPECIFIC_PROBLEM =
            AlarmIRPConstDefs::AttributeNameValue::SPECIFIC_PROBLEM;

        /**
         * This constant defines the name of the
         * correlatedNotifications property.
         * The data type for the value of this property
         * is AlarmIRPConstDefs::CorrelatedNotificationSet.
         */
        const string CORRELATED_NOTIFICATIONS =
            AlarmIRPConstDefs::AttributeNameValue::
                CORRELATED_NOTIFICATIONS;

        /**
         * This constant defines the name of the
         * backedUpStatus property.
         * The data type for the value of this property
         * is AlarmIRPConstDefs::BackedUpStatus.
         */
        const string BACKED_UP_STATUS =
            AlarmIRPConstDefs::AttributeNameValue::BACKED_UP_STATUS;

        /**
         * This constant defines the name of the backUpObject property.
         * The data type for the value of this property
```

```
    * is a string carrying of DN of the back-up object.
    */
   const string BACK_UP_OBJECT =
      AlarmIRPConstDefs::AttributeNameValue::BACK_UP_OBJECT;


   /**
    * This constant defines the name of the
    * trendIndication property.
    * The data type for the value of this property
    * is AlarmIRPConstDefs::TrendIndication.
    */
   const string TREND_INDICATION =
      AlarmIRPConstDefs::AttributeNameValue::TREND_INDICATION;


   /**
    * This constant defines the name of the thresholdInfo property.
    * The data type for the value of this property
    * is AlarmIRPConstDefs::ThresholdInfo.
    */
   const string THRESHOLD_INFO =
      AlarmIRPConstDefs::AttributeNameValue::THRESHOLD_INFO;


   /**
    * This constant defines the name of the
    * stateChangeDefinition property.
    * The data type for the value of this property
    * is AlarmIRPConstDefs::AttributeChangeSet.
    */
   const string STATE_CHANGE_DEFINITION =
      AlarmIRPConstDefs::AttributeNameValue::STATE_CHANGE_DEFINITION;


   /**
    * This constant defines the name of the
    * monitoredAttributes property.
    * The data type for the value of this property
    * is AlarmIRPConstDefs::AttributeSet.
    */
   const string MONITORED_ATTRIBUTES =
      AlarmIRPConstDefs::AttributeNameValue::MONITORED_ATTRIBUTES;


   /**
    * This constant defines the name of the
    * proposedRepairActions property.
    * The data type for the value of this property
    * is string.
    */
   const string PROPOSED_REPAIR_ACTIONS =
      AlarmIRPConstDefs::AttributeNameValue::PROPOSED_REPAIR_ACTIONS;


   /**
    * This constant defines the name of the additionalText
    * property.
    * The data type for the value of this property
    * is string.
    */
   const string ADDITIONAL_TEXT =
      AlarmIRPConstDefs::AttributeNameValue::ADDITIONAL_TEXT;


   /**
    * This constant defines the name of the alarmId property.
    * The data type for the value of this property
    * is 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.
    */
   const string ALARM_ID =
      AlarmIRPConstDefs::AttributeNameValue::ALARM_ID;


   /**
    * This constant defines the name of the serviceUser property.
    * The data type for the value of this property
    * is string.
    */
```

```
      const string SERVICE_USER =
         AlarmIRPConstDefs::AttributeNameValue::SERVICE_USER;

   /**
    * This constant defines the name of the serviceProvider
    * property.
    * The data type for the value of this property
    * is string.
    */
   const string SERVICE_PROVIDER =
      AlarmIRPConstDefs::AttributeNameValue::SERVICE_PROVIDER;

   /**
    * This constant defines the name of the
    * securityAlarmDetector property.
    * The data type for the value of this property
    * is string.
    */
   const string SECURITY_ALARM_DETECTOR =
      AlarmIRPConstDefs::AttributeNameValue::SECURITY_ALARM_DETECTOR;
};

interface NotifyAckStateChanged:
   NotificationIRPNotifications::Notify
{
   const string EVENT_TYPE = "notifyAckStateChanged";

   const string PROBABLE_CAUSE =
      AlarmIRPConstDefs::AttributeNameValue::PROBABLE_CAUSE;

   const string PERCEIVED_SEVERITY =
      AlarmIRPConstDefs::AttributeNameValue::PERCEIVED_SEVERITY;

   const string ALARM_ID =
      AlarmIRPConstDefs::AttributeNameValue::ALARM_ID;

   /**
    * This constant defines the name of the ackTime property.
    * The data type for the value of this property
    * is ManagedGenericIRPConstDefs::IRPTime.
    */
   const string ACK_TIME =
      AlarmIRPConstDefs::AttributeNameValue:: ACK_TIME;

   const string ACK_USER_ID =
      AlarmIRPConstDefs::AttributeNameValue::ACK_USER_ID;

   const string ACK_SYSTEM_ID =
      AlarmIRPConstDefs::AttributeNameValue::ACK_SYSTEM_ID;

   const string ACK_STATE =
      AlarmIRPConstDefs::AttributeNameValue::ACK_STATE;
};

interface NotifyClearedAlarm: NotificationIRPNotifications::Notify
{
   const string EVENT_TYPE = "notifyClearedAlarm";

   const string PROBABLE_CAUSE =
      AlarmIRPConstDefs::AttributeNameValue::PROBABLE_CAUSE;

   const string PERCEIVED_SEVERITY =
      AlarmIRPConstDefs::AttributeNameValue::PERCEIVED_SEVERITY;

   const string ALARM_ID =
      AlarmIRPConstDefs::AttributeNameValue::ALARM_ID;

   const string CLEAR_USER_ID =
      AlarmIRPConstDefs::AttributeNameValue::CLEAR_USER_ID;

   const string CLEAR_SYSTEM_ID =
      AlarmIRPConstDefs::AttributeNameValue::CLEAR_SYSTEM_ID;
};

interface NotifyAlarmListRebuilt:
   NotificationIRPNotifications::Notify
{
   const string EVENT_TYPE = "notifyAlarmListRebuilt";
```

```
        const string REASON =
            AlarmIRPConstDefs::AttributeNameValue::REASON;

        const string ALARM_LIST_ALIGNMENT_REQUIREMENT =
            AlarmIRPConstDefs::AttributeNameValue::
                ALARM_LIST_ALIGNMENT_REQUIREMENT;
    };

    interface NotifyChangedAlarm: NotificationIRPNotifications::Notify
    {
        const string EVENT_TYPE = "notifyChangedAlarm";

        const string PROBABLE_CAUSE =
            AlarmIRPConstDefs::AttributeNameValue::PROBABLE_CAUSE;

        const string PERCEIVED_SEVERITY =
            AlarmIRPConstDefs::AttributeNameValue::PERCEIVED_SEVERITY;

        const string ALARM_ID =
            AlarmIRPConstDefs::AttributeNameValue::ALARM_ID;
    };

    interface NotifyComments: NotificationIRPNotifications::Notify
    {
        const string EVENT_TYPE = "notifyComments";

        const string PROBABLE_CAUSE =
            AlarmIRPConstDefs::AttributeNameValue::PROBABLE_CAUSE;

        const string PERCEIVED_SEVERITY =
            AlarmIRPConstDefs::AttributeNameValue::PERCEIVED_SEVERITY;

        const string ALARM_ID =
            AlarmIRPConstDefs::AttributeNameValue::ALARM_ID;

        /**
         * This constant defines the name of the comments property.
         * The data type for the value of this property
         * is AlarmIRPConstDefs::CommentSet.
         */
        const string COMMENTS =
            AlarmIRPConstDefs::AttributeNameValue::COMMENTS;
    };

    interface NotifyPotentialFaultyAlarmList:
        NotificationIRPNotifications::Notify
    {
        const string EVENT_TYPE = "notifyPotentialFaultyAlarmList";

        /**
         * This constant defines the name of the reason property.
         * The data type for the value of this property
         * is string.
         */
        const string REASON =
            AlarmIRPConstDefs::AttributeNameValue::REASON;
    };

};

#endif // _ALARM_IRP_NOTIFICATIONS_IDL_
```

# Annex B (informative): Change history

| **Change history** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Date | TSG # | TSG Doc. | CR | Rev | Subject/Comment | Cat | Old | New |
| Dec 2006 | SA_34 | SP-060722 | 0049 | -- | Add filter complexity limitation parameter - Align with 32.111-2 | B | 6.6.0 | 7.0.0 |
| Mar 2007 | -- | -- | -- | -- | Delete reference to the 32.111-4 CMIP SS. Reason: SA#35 endorsed the SA5 decision to not propagate the CMIP Solution Sets to Rel-7 (TS 32.3x4, TS 32.4x4, TS 32.6x4) | -- | 7.0.0 | 7.0.1 |

# History

| Document history | | |
|---|---|---|
| V7.0.1 | March 2007 | Publication |
| | | |
| | | |
| | | |
| | | |