Source: **SA5 (Telecom Management)**

Title: **32.111 CR, "Split of TS - Part 3: Alarm Integration Reference Point (IRP): CORBA Solution Set (SS)" (S5-000333)**

Document for: **Approval**

Agenda Item: **6.5.3**

___

SA5 has split TS 32.111 into a multi-part TS 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".

Six (6) CRs are submitted to SA#8 for approval; the present one is highlighted in yellow:

Spec	CR	Phase	Subject	Cat	Version-Current	Version-New	Doc-2nd-Level
32.111	001	R99	Split of TS - Part 1: Main part of spec – Requirements	F	3.0.1	3.1.0	S5-000328
32.111	002	R99	Split of TS - Part 1: Main part of spec - Merging of Clause X into Clause 4, etc.	F	3.0.1	3.1.0	S5-000329
32.111	003	R99	Split of TS - Part 1: Main part of spec – Alignment of FM requirements with IRP, etc.	F	3.0.1	3.1.0	S5-000331
32.111	004	R99	Split of TS - Part 2: Alarm IRP Information Service (IS)	F	3.0.1	3.1.0	S5-000332
32.111	005	R99	Split of TS - Part 3: Alarm IRP CORBA Solution Set (SS)	F	3.0.1	3.1.0	S5-000333
32.111	006	R99	Split of TS - Part 4: Alarm IRP CMIP Solution Set (SS)	F	3.0.1	3.1.0	S5-000334

3G TS 32.111-3

Technical Specification

3rd Generation Partnership Project;
Technical Specification Group Services and System Aspects;
Part 3: Alarm Integration Reference Point:
CORBA Solution Set
(Release 1999)

Keywords
Fault Management, Alarms

3GPP

Postal address

3GPP support office address
650 Route des Lucioles - Sophia Antipolis
Valbonne - FRANCE
Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Internet
http://www.3gpp.org

Contents

Foreword

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

The present document is part 3 of multi-part 3G TS covering the 3rd Generation Partnership Project: Technical Specification Group Services and System Aspects, as identifies 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

This document specifies the CORBA Solution Set (SS) for the IRP whose semantics is specified in Alarm IRP: Information Service (IS) [9].

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 defines the usage of OMG CORBA Structured Event to carry information defined in notifications carrying alarm information. Clause 7 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.

[1]   ITU-T Recommendation X.721: "Information technology - Open Systems Interconnection - Structure of management information: Definition of management information".

[2]   ITU-T Recommendation X.736: "Information technology – Open Systems Interconnection – Security Alarm Reporting Function".

[3]   ITU-T Recommendation X.732: "Information technology – Open Systems Interconnection – Relationship Management Function".

[4]   ITU-T Recommendation X.732: "Information technology – Open Systems Interconnection – State Management Function".

[5]   ITU-T Recommendation X.732: "Information technology – Open Systems Interconnection – Object Management Function".

[6]   OMG Notification Service, OMG TC Document telecom/98-11-01

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

[8]   3G TS 32.106-8: Name Convention for Managed Objects

[9]   3G TS 32.111-2: Alarm IRP: Information Service, version 1

[10]   3G TS 32.106-2: Notification IRP: Information Service, version 1

[11]   3G TS 32.106-3: Notification IRP: CORBA Solution Set, version 1:1

[11]   ITU-T Recommendation X.735: "Information technology - Open Systems Interconnection - Systems Management: Log control function".

[12]   TS 32.111-1 3G Fault Management

[13]   TS 32.111-2 Alarm Integration Reference Point: Information Service

[14]   TS 32.111-4 Alarm Integration Reference Point: CMIP Solution Set

3   Definitions and abbreviations

3.1   Definitions

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

3.2   Abbreviations

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

CORBA	Common Object Request Broker Architecture
IDL	Interface Definition Language
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 Solution Set version

The version of this CORBA SS is 1:1, where the first "1" indicates the version number of the Alarm IRP: IS [9]; and the second "1" indicates the version number of this document.

4   Architectural Features

The overall architectural feature of Alarm IRP is specified in Reference [9]. 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 [6].

OMG Event Service provides event routing and distribution capabilities. OMG Notification Service provides, in addition to Event Service, event filtering and quality of service as well.

A necessary and sufficient sub set of OMG Notification Services shall be used to support AlarmIRPNotifications notifications as specified in [9].

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 and that support of pull style is optional.

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 must 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 [10].) 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 [6]. The name of the grammar is called "`EXTENDED_TCL`". See clause 2.4, Default Filter Constraint Language in [6]. This SS shall use this grammar only.

5 Mapping

5.1 Operation and Notification mapping

Alarm IRP: IS [9] defines semantics of operation and notification visible across the Alarm IRP. The table below 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 [9]	SS Method	Qualifier
acknowledgeAlarms	acknowledge_alarms	M
unacknowledgeAlarms	unacknowledge_alarms	O
getAlarmList	get_alarm_list	M
getAlarmIRPVersion	get_alarm_IRP_version	M
getAlarmCount	get_alarm_count	O
notifyNewAlarm	push_structured_event Note that OMG Notification Service [6] defines this method. See clause 8.1	M
notifyClearedAlarm	push_structured_event	M

	See clause 8.1	
notifyChangedAlarm	push_structured_event See clause 8.1	M
notifyAckStateChanged	push_structured_event See clause 8.1	M
notifyAlarmListRebuilt	push_structured_event See clause 8.1	M

5.2   Operation parameter mapping

Reference [9] defines semantics of parameters carried in operations across the Alarm IRP. Tables below 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
alarmInformation ReferenceList	AlarmIRPConstDefs::AlarmInformationIdSeq alarm_information_id_list	M
ackUserId	string ack_user_id	M
ackSystemId	string ack_system_id	O
bad AlarmInformation ReferenceList	AlarmIRPConstDefs::AlarmInformationIdSeq bad_alarm_information_id_list	M
status	CommonIRPConstDefs::Signal Exceptions: AcknowledgeAlarms, ParameterNotSupported, InvalidParameter	M

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

IS Operation parameter	SS Method parameter	Qualifier
alarm InformationReferen ceList	AlarmIRPConstDefs::AlarmInformationIdSeq alarm_information_id_list	M
ackUserId	string ack_user_id	M
ackSystemId	string ack_system_id	O
badAlarm Information ReferenceList	AlarmIRPConstDefs::AlarmInformationIdSeq bad_alarm_information_id_list	M
status	CommonIRPConstDefs::Signal Exceptions:	M

	UnacknowledgeAlarms, OperationNotSupported, ParameterNotSupported, InvalidParameter	

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, ParameterNotSupported, 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, OperationNotSupported, ParameterNotSupported, InvalidParameter	M

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

IS Operation parameter	SS Method parameter	Qualifier
versionNumberList	Return value of type CommonIRPConstDefs::VersionNumberSet	M
status	Exceptions: GetAlarmIRPVersion, InvalidParameter	M

5.3   Notification parameter mapping

Reference [9] defines semantics of parameters carried in notifications across the Alarm IRP.   Tables in this sub clause indicate the mapping of these parameters, as per notification, to their equivalents defined in this SS.

The following tables are relevant for notifyNewAlarm, notifyChangedAlarm, notifyClearedAlarm, notifyAckStateChanged.

Table 5: Mapping from IS `notify[New,Changed,Cleared]Alarm` and `notifyAckStateChanged` parameters to SS equivalents

IS Notification parameter	SS Notification parameter	Comment
notification Header	structuredEvent Note that OMG Notification Service [6] defines this structuredEvent. See Clause 4 as well.	Attributes of `notificationHeader` are mapped to attributes of `structuredEvent`. See clause 6.4 for attributes related to `notificationHeader`. See Table 9 for qualifiers for the parameter-attributes. For `notifyNewAlarm`, `notifyChangedAlarm`, `notifyClearedAlarm` and `notifyAckStateChanged`, the `extendedEventType` shall contain a string of `extendedEventTypeValue.NOTIFY_FM_NEW_ALARM`, `extendedEventTypeValue.NOTIFY_FM_CHANGED_ALARM`, `extendedEventTypeValue.NOTIFY_FM_CLEARED_ALARM`, `extendedEventTypeValue.NOTIFY_FM_ACK_STATE_CHANGED` respectively.
alarm Information Body	structuredEvent	Attributes of `alarmInformationBody` are mapped to attributes of `structuredEvent`. See clause 6.4 for attributes related to `alarmInformationBody`. See table 10 for qualifiers for the parameter-attributes.

The following table is relevant for `notifyAlarmListRebuilt`.

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

IS Notification parameter	SS equivalent	Comment
notification Header	structuredEvent	The `managedObjectClass`, `systemDN` shall be absent. The `eventType` shall contain a zero-length string. The `extendedEventType` shall contain a string of `extendedEventTypeValue.NOTIFY_FM_ALARM_LIST_REBUILT`. The `managedObjectInstance` shall carries the DN of the IRPAgent whose Alarm List has been rebuilt. Syntax and semantics of this string conform to the Managed Object string representation specified in [8]. See clause 6.4 for attributes related to `notificationHeader`. See Table 9 for qualifiers for the parameter-attributes.
reason	reason	It is a string indicating the Alarm List rebuilt reason

5.4 Parameter Attribute mapping

Notification IRP: IS [10] defines the semantics of attributes for `notificationHeader` parameter. Alarm IRP: IS [9] identifies `notificationHeader` for use for its IRP. Reference [9] also qualifies the attributes of the `notificationHeader` parameter. The following table shows the mapping of these IS attributes to SS equivalents.

Table 7: Mapping from IS `notificationHeader` attributes to SS equivalents

IS Attribute of `notificationHeader` in [10]	SS Attribute	Qualifier
managedObjectClass	managedObjectClass	O
managedObjectInstance	managedObjectInstance	M
notificationID	notificationID	M
eventTime	eventTime	M
systemDN	systemDN	M
eventType	eventType	M
extendedEventType	extendedEventType	M

Alarm IRP: IS [9] defines and qualifies the semantics of attributes for `alarmInformationBody` parameter. The following table shows the mapping of these IS attributes to SS equivalents.

Table 8: Mapping from IS `alarmInformationBody` attributes to SS equivalents

IS Attribute of `alarmInformationBody` in [9]	SS Attribute	Qualifier
probableCause	probableCause	M
perceivedSeverity	perceivedSeverity	M
specificProblem	specificProblem	O
correlatedNotifications	correlatedNotifications	O
backedUpStatus	backedUpStatus	O
backUpObject	backUpObject	O
trendIndication	trendIndication	O
thresholdInfo	thresholdInfo	O
stateChangeDefinition	stateChangeDefinition	O
monitoredAttributes	monitoredAttributes	O
proposedRepairActions	proposedRepairActions	O
additionalText	additionalText	O
additionalInformation.alarmId	alarmId	M
additionalInformation. ackTime	ackTime	note 1
additionalInformation. ackUserId	ackUserId	note 1
additionalInformation. ackSystemId	ackSystemId	note 1
additionalInformation. ackState	ackState	note 1

NOTE 1: See qualification information in [9], Table 13: Parameter-Attributes of `alarmInformationBody`.

6 Use of OMG Structured Event

Operation `notify` defined in [9] carries parameters, such as `notificationHeader` and `alarmInformationBody`. In CORBA SS, OMG defined `StructuredEvent` [2] is used to carry notification. This clause identifies the OMG defined `StructuredEvent` attributes that carry the attributes of parameters defined in [9].

The composition of OMG Structured Event, as defined in [6], is:

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

The table below lists all OMG Structured Event attributes in the second column. The first column identifies the SS attributes, if any, that shall be carried in the Structured Event attributes.

Table 9: Use of OMG Structured Event

SS Attribute	OMG CORBA Structured Event attribute	Comment
There is no corresponding SS attribute.	domain_name	It contains a string defined by interface IRPNotificationCategoryValue.alarmIRPVersion_1_1. It indicates the syntax and semantics of this Structured Event is defined by Alarm IRP: CORBA SS 1:1.
eventType	type_name	Attribute eventType is an attribute of notificationHeader. It shall indicate one of the following ITU-T defined semantics: communications alarm, processing error alarm, environmental alarm, quality of service alarm and equipment alarm. It is a string. See block of const string definitions starting with "ET_" in the IDL.
extendedEventType	event_name	Attribute extendedEventType is an attribute of notificationHeader. It shall identify one of the following: • notify a new alarm • notify changes in alarm state • notify changes in alarm acknowledgement state • notify alarm cleared • notify Alarm List has been successfully rebuilt It is a string. See block of const string definitions starting with "NOTIFY_FM_" in the IDL.

There is no corresponding SS attribute.	`variable Header`	
`managedObject Class`, `managedObjectI nstance`	One NV pair of `filterable_ body_fields`	NV stands for name-value pair. Order arrangement of NV pairs is not significant. The name of NV-pair is always encoded in string. They are attributes of `notificationHeader`. Name of NV pair is a string, `AttributeNameValue.managedObjectInstance`. Value of NV pair is a string. See corresponding table in Notification IRP: CORBA SS [1].
`notification Id`	One NV pair of `filterable_ body_fields`	It is an attribute of `notificationHeader`. Name of NV pair is a string, `AttributeNameValue.notificationId`. Value of NV pair is a long. See corresponding table in Notification IRP: CORBA SS [11].
`eventTime`	One NV pair of `filterable_ body_fields`	It is an attribute of `notificationHeader`. Name of NV pair is `AttributeNameValue.eventTime`. Value of NV pair is a `IRPTime`. See corresponding table in Notification IRP: CORBA SS [11].
`systemDN`	One NV pair of `filterable_ body_fields`	It is an attribute of `notificationHeader`. Name of NV pair is a string, `AttributeNameValue.systemDN`. Value of NV pair is a string. See corresponding table in Notification IRP: CORBA SS [11].
`probableCause`	One NV pair of `filterable_ body_fields`	It is an attribute of `alarmInformationBody`. Name of NV pair is a string, `AttributeNameValue.probableCause`. Value of NV pair is a short defined by `ProbableCauseValue`.
`perceived Severity`	One NV pair of `filterable_ body_fields`	It is an attribute of `alarmInformationBody`. Name of NV pair is a string, `AttributeNameValue.perceivedSeverity`. Value of NV pair is a short defined by `PerceivedSeverityValue`.
`specific Problem`	One NV pair of `filterable_ body_fields`	It is an attribute of `alarmInformationBody`. Name of NV pair is a string, `AttributeNameValue.specificProblem`. Value of NV pair is a string.

correlated Notifications	One NV pair of filterable_ body_fields	It is an attribute of alarmInformationBody. Name of NV pair is a string, AttributeNameValue.correlatedNotifications. Value of NV pair is a CorrelatedNotificationSetType.
backed UpStatus	One NV pair of filterable_ body_fields	It is an attribute of alarmInformationBody. Name of NV pair is a string, AttributeNameValue.backedUpStatus. Value of NV pair is a boolean BackedUpStatusType.
backUpObject	One NV pair of filterable_ body_fields	It is an attribute of alarmInformationBody. Name of NV pair is a string, AttributeNameValue.backedUpStatus. Value of NV pair is a string carrying of DN of the back-up object. See [8] for the DN string representation.
trend Indication	One NV pair of filterable_ body_fields	It is an attribute of alarmInformationBody. Name of NV pair is a string, AttributeNameValue.trendIndication. Value of NV pair is an enum TrendIndicationType.
thresholdInfo	One NV pair of filterable_ body_fields	It is an attribute of alarmInformationBody. Name of NV pair is a string, ParameterNameValue.thresholdInfo. Value of NV pair is an enum ThresholdIndicationType.
stateChange Definition	One NV pair of filterable_ body_fields	It is an attribute of alarmInformationBody. Name of NV pair is a string, AttributeNameValue.stateChangeDefinition. Value of NV pair is an AttributeChangeSetType.
monitored Attributes	One NV pair of filterable_ body_fields	It is an attribute of alarmInformationBody. Name of NV pair is a string, AttributeNameValue.monitoredAttributes. Value of NV pair is an AttributeSetType.
proposed RepairActions	One NV pair of filterable_ body_fields	It is an attribute of alarmInformationBody. Name of NV pair is a string, AttributeNameValue.proposedRepairActions. Value of NV pair is a string.

additional Text	One NV pair of `filterable_ body_fields`	It is an attribute of `alarmInformationBody`. Name of NV pair is a string, `AttributeNameValue.additionalText`. Value of NV pair is a string.
additional Information.al armId	One NV pair of `filterable_ body_fields`	It is an attribute of `alarmInformationBody`. Name of NV pair is a string, `AttributeNameValue.alarmId`. 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 `probableCuase` is encoded as a short, it shall be converted into string before concatenation. The resultant string shall not contain spaces.
additional Information. ackTime	One NV pair of `filterable_ body_fields`	It is an attribute of `notificationHeader`. Name of NV pair is a string, `AttributeNameValue.ackTime`. Value of NV pair is a `IRPTime`.
additional Information. ackUserId	One NV pair of `filterable_ body_fields`	It is an attribute of `alarmInformationBody`. Name of NV pair is a string, `AttributeNameValue.ackUserId`. Value of NV pair is a string.
additional Information. ackSystemId	One NV pair of `filterable_ body_fields`	It is an attribute of `alarmInformationBody`. Name of NV pair is a string, `AttributeNameValue.ackSystemId`. Value of NV pair is a string.
additional Information. ackState	One NV pair of `filterable_bo dy_fields`	It is an attribute of `alarmInformationBody`. Name of NV pair is a string, `AttributeNameValue.ackState`. Value of NV pair is a short defined by `AlarmAckState`.
There is no corresponding SS attribute.	`remaining_ body`	

7 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.

7.1 Method push (M)

```
module CosNotifyComm {

    …

    Interface SequencePushConsumer : NotifyPublish {
         void push_structured_events(

                in CosNotification::EventBatch notifications)

        raises( CosEventComm::Disconnected);

            …

    }; // SequencePushConsumer

    …

}; // CosNotifyComm
```

Notes:

♦ The push_structured_events method takes an input parameter of type EventBatch as defined in the OMG CosNotification module [6]. 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.

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

♦ 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.

♦ IRPAgent may push EventBatch with only one Structured Event.

Annex A (normative):
IDL specification

```
/* ## Module: AlarmConstDefs

This module contains commonly used definitions.
===============================================================================
 */

#ifndef  AlarmIRPConstDefs_idl
#define  AlarmIRPConstDefs_idl

#include "CosNotification.idl"

module AlarmIRPConstDefs {

  /*
  This block identifies all TMN ITU-T defined event types used by Alarm
  IRP of this version.  Their semantics are defined by ITU-T.  Their
  encodings for this version of Alarm IRP are defined here.  Other IRP
  documents, or other versions of Alarm IRP, shall identify their own
  ITU-T defined event types for their use.  They shall define their encodings
  as well.  Note all values are unique among themselves.  Other IRP documents
  can use the same values.
  */

  const string ET_COMMUNICATIONS_ALARM = "x1";
  const string ET_PROCESSING_ERROR_ALARM = "x2";
  const string ET_ENVIRONMENTAL_ALARM = "x3";
  const string ET_QUALITY_OF_SERVICE_ALARM = "x4";
  const string ET_EQUIPMENT_ALARM = "x5";


  /*
  This block identifies IRP defined, and not ITU-T defined, event types
  used by this Alarm IRP version.
  */

  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_CLEARED_ALARM = "x4";
  const string NOTIFY_FM_ALARM_LIST_REBUILT = "x5";

  /*
  It indicates if an object has a back up.
  True implies backep 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};

  /*
  It indicates if some observed condition is getting better, worse,
  or not changing.
```

```
    */

    enum TrendIndicationType {LessSevere, NoChange, MoreSevere};

    /*
    It is used in a notification to report a changed attribute value.
    */

    struct AttributeValueChangeType {
       string    attributeName;
       any       oldValue; // type depends on attribute
       any       newValue; // type depends on attribute
    };

    typedef sequence <AttributeValueChangeType> AttributeChangeSetType;

    /*
    It is used in a notification to report a changed attribute value.
    */

    struct AttributeValueType {
       string attributeName;
       any  value;    // type will depend on the attribute
    };

    typedef sequence <AttributeValueType> AttributeSetType;


    /*
    It indicates the level of severity.
    */

    enum PerceivedSeverityValue {
       Indeterminate, Critical, Major, Minor, Warning, Cleared
    };


    /*
    This block identifies the probable cause of a reported alarm.
    */

       const short PC_INDETERMINATE = 0;
       const short PC_ALARM_INDICATION_SIGNAL = 1;
       const short PC_CALL_SETUP_FAILURE = 2;
       const short PC_DEGRADED_SIGNAL_M3100 = 3;
       const short PC_FAR_END_RECEIVER_FAILURE = 4;
       const short PC_FRAMING_ERROR_M3100 = 5;
       const short PC_LOSS_OF_FRAME = 6;
       const short PC_LOSS_OF_POINTER = 7;
       const short PC_LOSS_OF_SIGNAL = 8;
       const short PC_PAYLOAD_TYPE_MISMATCH = 9;
       const short PC_TRANSMISSION_ERROR = 10;
       const short PC_REMOTE_ALARM_INTERFACE = 11;
       const short PC_EXCESSIVE_BIT_ERROR_RATE = 12;
       const short PC_PATH_TRACE_MISMATCH = 13;
       const short PC_UNAVAILABLE = 14;
       const short PC_SIGNAL_LABEL_MISMATCH = 15;
       const short PC_LOSS_OF_MULTI_FRAME = 16;
       const short PC_BACK_PLANE_FAILURE = 51;
       const short PC_DATA_SET_PROBLEM = 52;
       const short PC_EQUIPMENT_IDENTIFIER_DUPLICATION = 53;
       const short PC_EXTERNAL_DEVICE_PROBLEM = 54;
       const short PC_LINE_CARD_PROBLEM = 55;
       const short PC_MULTIPLEXER_PROBLEM_M3100 = 56;
       const short PC_NE_IDENTIFIER_DUPLICATION = 57;
       const short PC_POWER_PROBLEM_M3100 = 58;
```

```
    const short PC_PROCESSOR_PROBLEM_M3100 = 59;
    const short PC_PROTECTION_PATH_FAILURE = 60;
    const short PC_RECEIVER_FAILURE_M3100 = 61;
    const short PC_REPLACEABLE_UNIT_MISSING = 62;
    const short PC_REPLACEABLE_UNIT_TYPE_MISMATCH = 63;
    const short PC_SYNCHRONISATION_SOURCE_MISMATCH = 64;
    const short PC_TERMINAL_PROBLEM = 65;
    const short PC_TIMING_PROBLEM_M3100 = 66;
    const short PC_TRANSMITTER_FAILURE_M3100 = 67;
    const short PC_TRUNK_CARD_PROBLEM = 68;
    const short PC_REPLACEABLE_UNIT_PROBLEM = 69;
    const short PC_AIR_COMPRESSOR_FAILURE = 101;
    const short PC_AIR_CONDITIONING_FAILURE = 102;
    const short PC_AIR_DRYER_FAILURE = 103;
    const short PC_BATTERY_DISCHARGING = 104;
    const short PC_BATTERY_FAILURE = 105;
    const short PC_COMMERICAL_POWER_FAILURE = 106;
    const short PC_COOLING_FAN_FAILURE = 107;
    const short PC_ENGINE_FAILURE = 108;
    const short PC_FIRE_DETECTOR_FAILURE = 109;
    const short PC_FUSE_FAILURE = 110;
    const short PC_GENERATOR_FAILURE = 111;
    const short PC_LOW_BATTERY_THRESHOLD = 112;
    const short PC_PUMP_FAILURE_M3100 = 113;
    const short PC_RECTIFIER_FAILURE = 114;
    const short PC_RECTIFIER_HIGH_VOLTAGE = 115;
    const short PC_RECTIFIER_LOW_F_VOLTAGE = 116;
    const short PC_VENTILATION_SYSTEM_FAILURE = 117;
    const short PC_ENCLOSURE_DOOR_OPEN_M3100 = 118;
    const short PC_EXPLOSIVE_GAS = 119;
    const short PC_FIRE = 120;
    const short PC_FLOOD = 121;
    const short PC_HIGH_HUMIDITY = 122;
    const short PC_HIGH_TEMPERATURE = 123;
    const short PC_HIGH_WIND = 124;
    const short PC_ICE_BUILD_UP = 125;
    const short PC_LOW_FUEL = 127;
    const short PC_LOW_HUMIDITY = 128;
    const short PC_LOW_CABLE_PRESSURE = 129;
    const short PC_LOW_TEMPERATURE = 130;
    const short PC_LOW_WATER = 131;
    const short PC_SMOKE = 132;
    const short PC_TOXIC_GAS = 133;
    const short PC_STORAGE_CAPACITY_PROBLEM_M3100 = 151;
    const short PC_MEMORY_MISMATCH = 152;
    const short PC_CORRUPT_DATA_M3100 = 153;
    const short PC_OUT_OF_CPU_CYCLES = 154;
    const short PC_SOFTWARE_ENVIRONMENT_PROBLEM = 155;
    const short PC_SOFTWARE_DOWNLOAD_FAILURE = 156;
    const short PC_ADAPTER_ERROR = 301;
    const short PC_APPLICATION_SUBSYSTEM_FAILURE = 302;
    const short PC_BANDWIDTH_REDUCTION = 303;
    const short PC_COMMUNICATION_PROTOCOL_ERROR = 305;
    const short PC_COMMUNICATION_SUBSYSTEM_FAILURE = 306;
    const short PC_CONFIGURATION_OR_CUSTOMIZING_ERROR = 307;
    const short PC_CONGESTION = 308;
    const short PC_CPU_CYCLES_LIMIT_EXCEEDED = 310;
    const short PC_DATA_SET_OR_MODEM_ERROR = 311;
    const short PC_DTE_DCE_INTERFACE_ERROR = 313;
    const short PC_EQUIPMENT_MALFUNCTION = 315;
    const short PC_EXCESSIVE_VIBRATION = 316;
    const short PC_FILE_ERROR = 317;
    const short PC_HEATING_OR_VENTILATION_OR_COOLING_SYSTEM_PROBLEM = 321;
    const short PC_HUMIDITY_UNACCEPTABLE = 322;
    const short PC_INPUT_OUTPUT_DEVICE_ERROR = 323;
    const short PC_INPUT_DEVICE_ERROR = 324;
```

```
const short PC_LAN_ERROR = 325;
const short PC_LEAK_DETECTION = 326;
const short PC_LOCAL_NODE_TRANSMISSION_ERROR = 327;
const short PC_MATERIAL_SUPPLY_EXHAUSTED = 330;
const short PC_OUT_OF_MEMORY = 332;
const short PC_OUTPUT_DEVICE_ERROR = 333;
const short PC_PERFORMANCE_DEGRADED = 334;
const short PC_PRESSURE_UNACCEPTABLE = 336;
const short PC_QUEUE_SIZE_EXCEEDED = 339;
const short PC_RECEIVE_FAILURE = 340;
const short PC_REMOTE_NODE_TRANSMISSION_ERROR = 342;
const short PC_RESOURCE_AT_OR_NEARING_CAPACITY = 343;
const short PC_RESPONSE_TIME_EXCESSIVE = 344;
const short PC_RETRANSMISSION_RATE_EXCESSIVE = 345;
const short PC_SOFTWARE_ERROR = 346;
const short PC_SOFTWARE_PROGRAM_ABNORMALLY_TERMINATED = 347;
const short PC_SOFTWARE_PROGRAM_ERROR = 348;
const short PC_TEMPERATURE_UNACCEPTABLE = 350;
const short PC_THRESHOLD_CROSSED = 351;
const short PC_TOXIC_LEAK_DETECTED = 353;
const short PC_TRANSMIT_FAILURE = 354;
const short PC_UNDERLYING_RESOURCE_UNAVAILABLE = 356;
const short PC_VERSION_MISMATCH = 357;
const short PC_A_BIS_TO_BTS_INTERFACE_FAILURE = 501;
const short PC_A_BIS_TO_TRX_INTERFACE_FAILURE = 502;
const short PC_ANTENNA_PROBLEM = 503;
const short PC_BATTERY_BREAKDOWN = 504;
const short PC_BATTERY_CHARGING_FAULT = 505;
const short PC_CLOCK_SYNCHRONISATION_PROBLEM = 506;
const short PC_COMBINER_PROBLEM = 507;
const short PC_DISK_PROBLEM = 508;
const short PC_EXCESSIVE_RECEIVER_TEMPERATURE = 510;
const short PC_EXCESSIVE_TRANSMITTER_OUTPUT_POWER = 511;
const short PC_EXCESSIVE_TRANSMITTER_TEMPERATURE = 512;
const short PC_FREQUENCY_HOPPING_DEGRADED = 513;
const short PC_FREQUENCY_HOPPING_FAILURE = 514;
const short PC_FREQUENCY_REDEFINITION_FAILED = 515;
const short PC_LINE_INTERFACE_FAILURE = 516;
const short PC_LINK_FAILURE = 517;
const short PC_LOSS_OF_SYNCHRONISATION = 518;
const short PC_LOST_REDUNDANCY = 519;
const short PC_MAINS_BREAKDOWN_WITH_BATTERY_BACKUP = 520;
const short PC_MAINS_BREAKDOWN_WITHOUT_BATTERY_BACKUP = 521;
const short PC_POWER_SUPPLY_FAILURE = 522;
const short PC_RECEIVER_ANTENNA_FAULT = 523;
const short PC_RECEIVER_MULTICOUPLER_FAILURE = 525;
const short PC_REDUCED_TRANSMITTER_OUTPUT_POWER = 526;
const short PC_SIGNAL_QUALITY_EVALUATION_FAULT = 527;
const short PC_TIMESLOT_HARDWARE_FAILURE = 528;
const short PC_TRANSCEIVER_PROBLEM = 529;
const short PC_TRANSCODER_PROBLEM = 530;
const short PC_TRANSCODER_OR_RATE_ADAPTER_PROBLEM = 531;
const short PC_TRANSMITTER_ANTENNA_FAILURE = 532;
const short PC_TRANSMITTER_ANTENNA_NOT_ADJUSTED = 533;
const short PC_TRANSMITTER_LOW_VOLTAGE_OR_CURRENT = 535;
const short PC_TRANSMITTER_OFF_FREQUENCY = 536;
const short PC_DATABASE_INCONSISTENCY = 537;
const short PC_FILE_SYSTEM_CALL_UNSUCCESSFUL = 538;
const short PC_INPUT_PARAMETER_OUT_OF_RANGE = 539;
const short PC_INVALID_PARAMETER = 540;
const short PC_INVALID_POINTER = 541;
const short PC_MESSAGE_NOT_EXPECTED = 542;
const short PC_MESSAGE_NOT_INITIALISED = 543;
const short PC_MESSAGE_OUT_OF_SEQUENCE = 544;
const short PC_SYSTEM_CALL_UNSUCCESSFUL = 545;
const short PC_TIMEOUT_EXPIRED = 546;
```

```
        const short PC_VARIABLE_OUT_OF_RANGE = 547;
        const short PC_WATCH_DOG_TIMER_EXPIRED = 548;
        const short PC_COOLING_SYSTEM_FAILURE = 549;
        const short PC_EXTERNAL_EQUIPMENT_FAILURE = 550;
        const short PC_EXTERNAL_POWER_SUPPLY_FAILURE = 551;
        const short PC_EXTERNAL_TRANSMISSION_DEVICE_FAILURE = 552;
        const short PC_REDUCED_ALARM_REPORTING = 561;
        const short PC_REDUCED_EVENT_REPORTING = 562;
        const short PC_RECUCED_LOGGING_CAPABILITY = 563;
        const short PC_SYSTEM_RESOURCES_OVERLOAD = 564;
        const short PC_BROADCAST_CHANNEL_FAILURE = 565;
        const short PC_CALL_ESTABLISHMENT_ERROR = 566;
        const short PC_INVALID_MESSAGE_RECEIVED = 567;
        const short PC_INVALID_MSU_RECEIVED = 568;
        const short PC_LAPD_LINK_PROTOCOL_FAILURE = 569;
        const short PC_LOCAL_ALARM_INDICATION = 570;
        const short PC_REMOTE_ALARM_INDICATION = 571;
        const short PC_ROUTING_FAILURE = 572;
        const short PC_SS7_PROTOCOL_FAILURE = 573;
        const short PC_TRANSMISSION_FAILURE = 574;


    typedef sequence <string> AlarmInformationIdSeq;

    typedef CosNotification::EventBatch AlarmInformationSeq;

    enum AlarmAckState {
       ActiveAndAcknowledged, ActiveAndUnacknowledged, ClearedAndUnacknowledged,
       ClearedAndUnacknowledged, All
    };

};

#endif


/* ## Module: AlarmIRPSystem

This module contains the specification of all operations of Alarm IRP Agent
specified in Alarm IRP: IS version 1 and Alarm IRP: CORBA SS version 1:1.
===============================================================================
*/

#ifndef  AlarmIRPSystem_idl
#define  AlarmIRPSystem_idl

#include "CosNotification.idl"
#include "AlarmIRPConstDefs.idl"
#include "CommonIRPConstDefs.idl"

module AlarmIRPSystem {

  /*
  System fails to complete the operation.  System provides
  reasons whose semantics is outside the scope of this IRP.
  */

  exception AcknowledgeAlarms { string reason; };
  exception UnacknowledgeAlarms { string reason; };
  exception GetAlarmList {   string reason; };
  exception GetAlarmIRPVersion { string reason; };
  exception GetAlarmCount { string reason; };
  exception ParameterNotSupported { string parameter; };
    //name of the unsupported parameter as defined in IDL.
  exception InvalidParameter { string parameter; };
    //name of the parameter as defined in IDL
```

```
    exception OperationNotSupported {};
    exception NextAlarmInformations { string reason; };


       /**
       The AlarmInformationIterator is used to iterate through a set of Alarm
       Informations in Alarm List.  Method get_alarm_list contains it as
        output parameter.
       IRPManager uses it to pace the return of Alarm Informations. IRPManager
        cannot use it to pace when IRPAgent should retrieve Alarm Informations
        from Alarm List.
       */

       interface AlarmInformationIterator {

           /**
           This method returns up to "how_many" Alarm Informations.
           If 1 or more Alarm Information is returned, return TRUE.
           Return FALSE if there is no more Alarm Information to be returned.
           */

           boolean next_alarmInformations (
             in unsigned long how_many,
             out AlarmIRPConstDefs::AlarmInformationSeq alarm_informations
           )
           raises (NextAlarmInformations,InvalidParameter);

           /**
           This method destroys the iterator.
           */

           void destory ();

       }; // end of AlarmInformationIterator


    /*
    This interface specifies all methods supported by System as
    specified in 3GPP AlarmIRP: CORBA Solution Set version 1:1.
    */

    interface AlarmIRPOperations {

      CommonIRPConstDefs::Signal acknowledge_alarms (
         in AlarmIRPConstDefs::AlarmInformationIdSeq alarm_information_id_list,
         in string ack_user_id,
         in string ack_system_id,
         out AlarmIRPConstDefs::AlarmInformationIdSeq
        bad_alarm_information_id_list
      )
      raises (AcknowledgeAlarms,ParameterNotSupported,InvalidParameter);


      CommonIRPConstDefs::Signal unacknowledge_alarms (
         in AlarmIRPConstDefs::AlarmInformationIdSeq alarm_information_id_list,
         in string ack_user_id,
         in string ack_system_id,
         out AlarmIRPConstDefs::AlarmInformationIdSeq
           bad_alarm_information_id_list
      )
      raises (UnacknowledgeAlarms, OperationNotSupported, ParameterNotSupported,
           InvalidParameter);


       /*
```

```
     This method returns Alarm Informations.
     If flag is TRUE, all returned Alarm Informations shall be
     in AlarmInformationSeq that contains 0,1 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,ParameterNotSupported,InvalidParameter);


     void get_alarm_count (
        in string filter,
        out long critical_count,
        out long major_count,
        out long minor_count,
        out long warning_count,
        out long indeterminate_count,
        out long cleared_count
     )
    raises (GetAlarmCount, OperationNotSupported, ParameterNotSupported,
         InvalidParameter);


     CommonIRPConstDefs::VersionNumberSet get_alarm_IRP_version ()
        raises (GetAlarmIRPVersion, InvalidParameter);
   };

};

#endif
```

Annex B (informative):
Change history

Change history					
TSG SA#	Version	CR	Tdoc SA	New Version	Subject/Comment
-	0.0.0	-	-	-	-

<table>
<tr><td colspan="3">CHANGE REQUEST</td><td><i>Please see embedded help file at the bottom of this page for instructions on how to fill in this form correctly.</i></td></tr>
</table>

32.111	CR	**005**	Current Version:	3.0.1

GSM (AA.BB) or 3G (AA.BBB) specification number ↑   ↑ *CR number as allocated by MCC support team*

For submission to:	SA#8	for approval	X	strategic	
list expected approval meeting # here ↑		for information		non-strategic	
					(for SMG use only)

Form: CR cover sheet, version 2 for 3GPP and SMG   *The latest version of this form is available from: ftp://ftp.3gpp.org/Information/CR-Form-v2.doc*

Proposed change affects: (U)SIM [ ]   ME [ ]   UTRAN / Radio [X]   Core Network [ ]
(at least one should be marked with an X)

Source:	SA5#12	**Date:**	20 June 2000

Subject:	Split of TS - Part 3: Alarm Integration Reference Point (IRP): CORBA Solution Set (SS)

Work item:	32.111 3G Fault Management

Category: F Correction   [X]   **Release:**   Phase 2 [ ]

(only one category shall be marked with an X)

Category		Release	
A	Corresponds to a correction in an earlier release	Release 96	
B	Addition of feature	Release 97	
C	Functional modification of feature	Release 98	
D	Editorial modification	Release 99	X
		Release 00	

Reason for change:	The following changes are proposed to be introduced in TS 32.111 Ver 3.0.1

The change contains the TS 32.111 Part-3 (Alarm IRP CORBA Solution Set)

This part is not present in the version 3.0.1 of TS 32.111, therefore the entire document must be seen as "new". |

Clauses affected:	None of the current TS 32.111 clauses are affected.

Other specs affected:

Other 3G core specifications	[ ]	→ List of CRs:	
Other GSM core specifications	[ ]	→ List of CRs:	
MS test specifications	[ ]	→ List of CRs:	
BSS test specifications	[ ]	→ List of CRs:	
O&M specifications	[ ]	→ List of CRs:	

Other comments:	