Technical Specification Group Services and System Aspects
Meeting #15, Cheju Island, Korea, 11-14 March 2002

TSGS#15(02)0028

Source:	**SA5 (Telecom Management)**
Title:	**3 Rel-4 CR 32.111-2/3/4 (Fault Management; Alarm Integration Reference Point; Part 2 Information Service/ Part 3 CORBA SS/ Part 4 CMIP SS) Addition of "perceivedSeverity" as parameter to "acknowledgeAlarms" operation**
Document for:	**Decision**
Agenda Item:	**7.5.3**

The "children" CRs 32.111-3CR015 and 32.111-4CR005 should only be approved if their "parent" CR 32.111-2CR012 was approved.

Doc-1st-Level	Spec	CR	R	Phase	Subject	Cat	Version-Current	Version-New	Doc-2nd-Level	Workitem
SP-020028	32.111-2	012		Rel-4	**Addition of "perceivedSeverity" as parameter to "acknowledgeAlarms operation" (IS)**	F	4.2.0	4.3.0	S5-010769	OAM-FM
SP-020028	32.111-3	015		Rel-4	**Addition of "perceivedSeverity" as parameter to "acknowledgeAlarms" operation (CORBA SS)**	F	4.1.0	4.2.0	S5-010770	OAM-FM
SP-020028	32.111-4	005		Rel-4	**Addition of "perceivedSeverity" as parameter to "acknowledgeAlarms" operation (CMIP SS)**	F	4.1.0	4.2.0	S5-010771	OAM-FM

3GPP TSG-SA5 (Telecom Management) **S5-010769**
Meeting #23, Sundsvall, SWEDEN, 15 – 19 October 2001 **S5C010413rev2**

CR-Form-v4

CHANGE REQUEST

⌘	**32.111-2**	**CR**	**012**	⌘	ev	**-**	⌘	Current version:	**4.2.0** ⌘

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

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

Title:	⌘	Addition of "perceivedSeverity" as parameter to "acknowledgeAlarms operation" (IS)
Source:	⌘	SA5
Work item code:	⌘	OAM-FM
Date:	⌘	01/03/2002
Category:	⌘	**F**
Release:	⌘	REL-4

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

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change:	⌘	Assumption: an acknowledgeAlarms operation is initiated at the manager side at approx. the same time as the agent is notifying the manager of a change in severity, both related to the same alarm id – this is leading to a mismatch between the agents and the managers alarm list.
Summary of change:	⌘	Add perceivedSeverity as parameter to acknowledgeAlarms operation.
Consequences if not approved:	⌘	Allowing for error in agent alarm list (mismatch between the agents and the managers alarm list).

Clauses affected:	⌘	6.3.1.2, 6.3.1.3, 6.3.1.4			
Other specs affected:	⌘	□	Other core specifications	⌘	
		□	Test specifications		
		X	O&M Specifications		If this "parent" CR 32.111-2CR012 is approved, then also the attached "children" CRs can be approved: 32.111-3CR015 (S5-010770), 32.111-4CR005 (S5-010771).
Other comments:	⌘	None			

6.3.1 acknowledgeAlarms (M)

6.3.1.1 Definition

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

6.3.1.2 Input Parameters

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

6.3.1.3 Output Parameters

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

			If operation_failed is true, status = OperationFailed.

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

6.3.1.4 Pre-condition

`atLeastOneValidId.`

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

6.3.1.5 Post-condition

`someAlarmAcknowledged OR allAlarmsAcknowledged.`

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

6.3.1.6 Exceptions

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

3GPP TSG-SA5 (Telecom Management)
Meeting #24, Cancun, MEXICO, 26 – 30 November 2001

S5-010770
S5C010549rev6

CR-Form-v4

CHANGE REQUEST

⌘ **32.111-3** **CR** **CRNum** ⌘ ev **-** ⌘ Current version: **4.1.0** ⌘

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

Proposed change affects: ⌘ (U)SIM □ ME/UE □ Radio Access Network ☒ Core Network ☒

Title:	⌘	Addition of "perceivedSeverity" as parameter to "acknowledgeAlarms" operation (CORBA SS)
Source:	⌘	SA5
Work item code:	⌘	OAM-FM
Date:	⌘	01/03/2002
Category:	⌘	F
Release:	⌘	REL-4

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

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change:	⌘	An acknowledgeAlarms operation and a notification of a change in severity occuring concurrently creates a mismatch between the agents and the managers alarm list.
Summary of change:	⌘	Update Corba SS in accordance with corrected IS (added perceivedSeverity as parameter to acknowledgeAlarms operation).
Consequences if not approved:	⌘	Risk for mismatch between the agents and the managers alarm list.

Clauses affected:	⌘	5.2, Annex A		
Other specs affected:	⌘	□ Other core specifications □ Test specifications ☒ O&M Specifications	⌘	This "child" CR should only be approved if its attached "parent" CR 32.111-2CR012 (S5-010769) was approved.
Other comments:	⌘	None		

How to create CRs using this form:

Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://ftp.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2001-03 contains the specifications resulting from the March 2001 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

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 indicate the mapping of these parameters, as per operation, to their equivalents defined in this SS.

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

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

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

```
#ifndef AlarmIRPConstDefs_idl
#define AlarmIRPConstDefs_idl

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

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

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

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

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

   /*
   This block identifies the notification types defined by this
   Alarm IRP version.
   */
   interface NotificationType
   {
      const string NOTIFY_FM_NEW_ALARM = "x1";
      const string NOTIFY_FM_CHANGED_ALARM = "x2";
      const string NOTIFY_FM_ACK_STATE_CHANGED = "x3";
      const string NOTIFY_FM_COMMENT_ADDED = "x4";
      const string NOTIFY_FM_CLEARED_ALARM = "x5";
      const string NOTIFY_FM_ALARM_LIST_REBUILT = "x6";
   };

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

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

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

```
      const short RETRANSMISSION_RATE_EXCESSIVE = 345;
      const short SOFTWARE_ERROR = 346;
      const short SOFTWARE_PROGRAM_ABNORMALLY_TERMINATED = 347;
      const short SOFTWARE_PROGRAM_ERROR = 348;
      const short TEMPERATURE_UNACCEPTABLE = 350;
      const short THRESHOLD_CROSSED = 351;
      const short TOXIC_LEAK_DETECTED = 353;
      const short TRANSMIT_FAILURE = 354;
      const short UNDERLYING_RESOURCE_UNAVAILABLE = 356;
      const short VERSION_MISMATCH = 357;
      const short A_BIS_TO_BTS_INTERFACE_FAILURE = 501;
      const short A_BIS_TO_TRX_INTERFACE_FAILURE = 502;
      const short ANTENNA_PROBLEM = 503;
      const short BATTERY_BREAKDOWN = 504;
      const short BATTERY_CHARGING_FAULT = 505;
      const short CLOCK_SYNCHRONISATION_PROBLEM = 506;
      const short COMBINER_PROBLEM = 507;
      const short DISK_PROBLEM = 508;
      const short EXCESSIVE_RECEIVER_TEMPERATURE = 510;
      const short EXCESSIVE_TRANSMITTER_OUTPUT_POWER = 511;
      const short EXCESSIVE_TRANSMITTER_TEMPERATURE = 512;
      const short FREQUENCY_HOPPING_DEGRADED = 513;
      const short FREQUENCY_HOPPING_FAILURE = 514;
      const short FREQUENCY_REDEFINITION_FAILED = 515;
      const short LINE_INTERFACE_FAILURE = 516;
      const short LINK_FAILURE = 517;
      const short LOSS_OF_SYNCHRONISATION = 518;
      const short LOST_REDUNDANCY = 519;
      const short MAINS_BREAKDOWN_WITH_BATTERY_BACKUP = 520;
      const short MAINS_BREAKDOWN_WITHOUT_BATTERY_BACKUP = 521;
      const short POWER_SUPPLY_FAILURE = 522;
      const short RECEIVER_ANTENNA_FAULT = 523;
      const short RECEIVER_MULTICOUPLER_FAILURE = 525;
      const short REDUCED_TRANSMITTER_OUTPUT_POWER = 526;
      const short SIGNAL_QUALITY_EVALUATION_FAULT = 527;
      const short TIMESLOT_HARDWARE_FAILURE = 528;
      const short TRANSCEIVER_PROBLEM = 529;
      const short TRANSCODER_PROBLEM = 530;
      const short TRANSCODER_OR_RATE_ADAPTER_PROBLEM = 531;
      const short TRANSMITTER_ANTENNA_FAILURE = 532;
      const short TRANSMITTER_ANTENNA_NOT_ADJUSTED = 533;
      const short TRANSMITTER_LOW_VOLTAGE_OR_CURRENT = 535;
      const short TRANSMITTER_OFF_FREQUENCY = 536;
      const short DATABASE_INCONSISTENCY = 537;
      const short FILE_SYSTEM_CALL_UNSUCCESSFUL = 538;
      const short INPUT_PARAMETER_OUT_OF_RANGE = 539;
      const short INVALID_PARAMETER = 540;
      const short INVALID_POINTER = 541;
      const short MESSAGE_NOT_EXPECTED = 542;
      const short MESSAGE_NOT_INITIALISED = 543;
      const short MESSAGE_OUT_OF_SEQUENCE = 544;
      const short SYSTEM_CALL_UNSUCCESSFUL = 545;
      const short TIMEOUT_EXPIRED = 546;
      const short VARIABLE_OUT_OF_RANGE = 547;
      const short WATCH_DOG_TIMER_EXPIRED = 548;
      const short COOLING_SYSTEM_FAILURE = 549;
      const short EXTERNAL_EQUIPMENT_FAILURE = 550;
      const short EXTERNAL_POWER_SUPPLY_FAILURE = 551;
      const short EXTERNAL_TRANSMISSION_DEVICE_FAILURE = 552;
      const short REDUCED_ALARM_REPORTING = 561;
      const short REDUCED_EVENT_REPORTING = 562;
      const short RECUCED_LOGGING_CAPABILITY = 563;
      const short SYSTEM_RESOURCES_OVERLOAD = 564;
      const short BROADCAST_CHANNEL_FAILURE = 565;
      const short CALL_ESTABLISHMENT_ERROR = 566;
      const short INVALID_MESSAGE_RECEIVED = 567;
      const short INVALID_MSU_RECEIVED = 568;
      const short LAPD_LINK_PROTOCOL_FAILURE = 569;
      const short LOCAL_ALARM_INDICATION = 570;
      const short REMOTE_ALARM_INDICATION = 571;
      const short ROUTING_FAILURE = 572;
      const short SS7_PROTOCOL_FAILURE = 573;
      const short TRANSMISSION_FAILURE = 574;
   };

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

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

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

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

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

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


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


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

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

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

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

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


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

   @member attributeID: identifies the attribute that
     crossed the threshold.
   @member observedValue: attributes that are of type
     integer will be converted to floats.
   @member thresholdlevel: This parameter is for
     multi-level threhsolds. Optional.
   @member armTime: May contain empty string.
*/

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


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

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

   typedef sequence <AttributeValueChangeType> AttributeChangeSetType;

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

   typedef sequence <AttributeValueType> AttributeSetType;

   typedef sequence <long> NotifIdSetType;

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

   /*
   Correlated Notification sets are sets of Correlated Notification
   structures.
   */
   typedef sequence <CorelatedNotification> CorrelatedNotificationSetType;
   /*
   Define the structure of Alarm ID and Perceived Severity used within the
   alarm acknowledgment operation. Note: perceivedSeverity is an optional
   parameter.
   */
   struct AlarmInformationIdAndSev
   {
      string alarm_information_reference;
      PerceivedSeverity perceived_severity;
   };

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

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

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

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

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

IDL specification (file name “AlarmIRPSystem.idl”)
#ifndef AlarmIRPSystem_idl
#define AlarmIRPSystem_idl
```

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

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

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

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

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

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

      /*
      This method destroys the iterator.
      */
      void destroy();
   };
   interface AlarmIRP
   {
      /*
      Return the list of all supported Alarm IRP versions.
      */
      ManagedGenericIRPConstDefs::VersionNumberSet get_alarm_IRP_versions (
      )
      raises (GetAlarmIRPVersions);


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

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


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


      /*
      Request to remove acknowledgement information of one or more alarms.
      */
      ManagedGenericIRPConstDefs::Signal unacknowledge_alarms (
         in AlarmIRPConstDefs::AlarmInformationIdSeq alarm_information_id_list,
         in string ack_user_id,
         in string ack_system_id,
         out AlarmIRPConstDefs::BadAlarmInformationIdSeq
             bad_alarm_information_id_list
      )
      raises (UnacknowledgeAlarms,
              ManagedGenericIRPSystem::OperationNotSupported,
              ManagedGenericIRPSystem::ParameterNotSupported,
                ManagedGenericIRPSystem::InvalidParameter);


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

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


      /*
      This method returns the count of Alarm Informations.
      */
      void get_alarm_count (
         in string filter,
         out unsigned long critical_count,
         out unsigned long major_count,
         out unsigned long minor_count,
         out unsigned long warning_count,
         out unsigned long indeterminate_count,
         out unsigned long cleared_count
      )
      raises (GetAlarmCount, ManagedGenericIRPSystem::OperationNotSupported,
              ManagedGenericIRPSystem::ParameterNotSupported,
              ManagedGenericIRPSystem::InvalidParameter);
   };
};

#endif
```

3GPP TSG-SA5 (Telecom Management) **S5-010771**
Meeting #24, Cancun, MEXICO, 26 – 30 Novenmber 2001 **S5C010558rev2**

CR-Form-v4

CHANGE REQUEST

⌘	**32.111-4**	**CR**	**005**	⌘	ev	**-**	⌘	Current version:	**4.0.0** ⌘

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

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

Title:	⌘	Addition of "perceivedSeverity" as parameter to "acknowledgeAlarms" operation (CMIP SS)		
Source:	⌘	SA5		
Work item code:	⌘	OAM-FM	***Date:*** ⌘	01/03/2002
Category:	⌘	**F**	***Release:*** ⌘	REL-4

Use one of the following categories:
__F__ (correction)
__A__ (corresponds to a correction in an earlier release)
__B__ (addition of feature),
__C__ (functional modification of feature)
__D__ (editorial modification)
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change:	⌘	Alignment of CMIP SS (32.111-3) with IS (32.111-2) according to IS changes regarding added perceivedSeverity as parameter to acknowledgeAlarms.
Summary of change:	⌘	Correct the mapping table with respect to alarmInformationAndSeverityReferenceList.
Consequences if not approved:	⌘	Mismatch between the IS and the CMIP SS.

Clauses affected:	⌘	4.7.3			
Other specs affected:	⌘	☐	Other core specifications	⌘	
		☐	Test specifications		
		☒	O&M Specifications		This "child" CR should only be approved if its attached "parent" CR 32.111-2CR012 (S5-010769) was approved.
Other comments:	⌘				

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://ftp.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2001-03 contains the specifications resulting from the March 2001 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

4.7.3 Mapping of Parameters of each operation

The tables in the following clauses show the parameters of each operations defined in the IS 3GPP TS 32.111-2 [9] and their equivalents in this CMIP SS.

The input parameters of the operations are mapped into "Action information" (see GDMO and ASN.1 definitions for more details).

The output parameters of the operations are mapped into "Action response" (see GDMO and ASN.1 definitions for more details).

Table 2: Mapping of parameters of 'acknowledgementAlarms'

Operation parameters of Information Services	IN/OUT	CMIP equivalences	Qualifier
alarmInformationAndSeverityReferenceList	IN	AlarmReferenceList (Note: severity verification not required in CMIP solution set)	M
ackUserId	IN	ackUserId	M
ackSystemId	IN	ackSystemId	O
badAlarmInformationReferenceList	OUT	errorAlarmReferenceList	M
status	OUT	status	M