Source: **SA5 (Telecom Management)**

Title: **3 Rel-6 CR 32.111-2/3/4 (Fault Management IS/ CORBA/ CMIP SS) : Align the operation getAlarmList with the notification notifyAlarmListRebuilt**

Document for: **Decision**

Agenda Item: **7.5.3**

Doc-1st-Level	Spec	CR	Ph	Subject	Cat	Ver-Cur	Doc-2nd-Level	WI
SP-030629	32.111-2	027	Rel-6	Align the operation getAlarmList with the notification notifyAlarmListRebuilt	B	5.4.0	S5-037277	OAM-NIM
SP-030629	32.111-3	033	Rel-6	Align operation getAlarmList with the notification notifyAlarmListRebuilt	B	5.4.0	S5-037255	OAM-NIM
SP-030629	32.111-4	025	Rel-6	Align operation getAlarmList with the notification notifyAlarmListRebuilt	B	5.6.0	S5-037256	OAM-NIM

3GPP TSG-SA5 (Telecom Management) **S5-037277**
Meeting #36, Shanghai, CHINA, 17 - 21 Nov 2003

CR-Form-v7

CHANGE REQUEST

⌘ **32.111-2** **CR** **027** ⌘ **rev** **-** ⌘ Current version: **5.4.0** ⌘

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

Proposed change affects: UICC apps⌘ ☐ ME ☐ Radio Access Network ☒ Core Network ☒

Title:	⌘	Align the operation *getAlarmList* with the notification *notifyAlarmListRebuilt*
Source:	⌘	**SA5** (olaf.pollakowski@siemens.com)
Work item code:	⌘	OAM-NIM
Date:	⌘	21/11/2003
Category:	⌘	**B**
Release:	⌘	Rel-6

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)*
Rel-6	*(Release 6)*

Reason for change:	⌘	The notification notifyAlarmListRebuilt can indicate that only part of the alarm list has been rebuilt. The manager is unable to react to this accordingly by requesting the upload of the part of the alarm list that has been rebuilt.
Summary of change:	⌘	The CR adds the parameters objectClass and objectInstance to the input parameter list of getAlarmList. These parameters are required to identify the part of the alarm list to be uploaded to the manager in case of a partial alarm alignment.
Consequences if not approved:	⌘	The functionality of notifyAlarmList and getAlarmList is not aligned.

Clauses affected:	⌘	6.3.2

		Y	N			
Other specs affected:	⌘		X	Other core specifications	⌘	
			X	Test specifications		
		X		O&M Specifications		32.111-3, 32.111-4

Other comments:	⌘	Parent

Change in Clause 6.3.2

6.3.2 getAlarmList (M)

6.3.2.1 Definition

The `IRPManager` invokes this operation in order to request~~s~~ the `IRPAgent` to provide either the complete list of `AlarmInformation` instances in the `AlarmList`, including (when supported) the IOC instances associated with the `AlarmInformation` instances (full alarm alignment), or only a part of this list (partial alarm alignment).

The parameters `baseObjectClass` and `baseObjectInstance` are used to identify the part of the alarm list to be returned. If they are absent, then the complete alarm list shall be provided (full alarm alignment). If they identify a certain MO, then only the `AlarmInformation` instances (and associated IOC instances) related to this MO and its subordinate MOs shall be provided (partial alarm alignment).

There are two modes of operation. One mode is synchronous. In this mode, the list of `AlarmInformation instances` in `AlarmList` is returned synchronously with the operation. The other mode is asynchronous. In this mode, the list of `AlarmInformation` instances is returned via notifications. In asynchronous mode of operation, the only information returned synchronously is the status of the operation. The mode of operation to be used is determined by means outside the scope of specification. To use asynchronous mode, the `IRPManager` must have established a subscription with the IRPAgent notificationIRP via the `subscribe` operation specified in [**Error! Reference source not found.**].

6.3.2.2 Input Parameters

Name	Qualifier	Information Type	Comment
alarmAckState	O	ENUM (all alarms, all active alarms, all active and acknowledged alarms, all active and unacknowledged, all Cleared and unacknowledged alarms, all unacknowledged)	It carries a constraint. The IRPAgent shall apply it on AlarmInformation instances in AlarmList when constructing its output parameter AlarmInformationList.
baseObjectClass	O, see note 1	This parameter is either absent or carries the object class of a certain MO.	If this paramter is absent, then all AlarmInformation instances in the AlarmList shall be returned. If the paramter carries the object class of a certain MO, then all AlarmInformation instances (and associated IOC instances) of the MO identified by the parameter baseObjectInstance and its subordinate MOs shall be returned. The AlarmInformation instances not related to the subject MO and its subordinate MOs shall not be returned (see note 2).
baseObjectInstance	O, see note 1	This parameter is either absent or carries the DN of a certain MO.	If the objectClass parameter is absent, then this parameter shall be absent. If the baseObjectClass parameter carries the object class of a certain MO, then this parameter shall carry the DN of the related MO instance. The AlarmList has to be returned only for alarms concerning that MO and its subordinate MOs (see note 2).
filter	O	N/A	It carries a filter constraint. The IRPAgent shall apply it on AlarmInformation instances in AlarmList when constructing its output parameter AlarmInformationList.

Note 1: If the notification notifyAlarmListRebuilt supports indicating that only a part of the alarm list has been rebuilt then the operation getAlarmList shall support partial alarm alignment.

Note 2: The legal values of the parameters baseObjectClass and baseObjectInstance are restricted to those carried by the parameters baseObjectClass and baseObjectInstance in the recent notifyAlarmListRebuilt notifications. The timeline for "recent" is vendor-specific.

6.3.2.3 Output Parameters

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

		(OperationSucceeded, OperationFailed)	OperationSucceeded. If operation_failed is true, status = OperationFailed.

6.3.2.4 Pre-condition

baseObjectExists

Assertion Name	Definition
baseObjectExists	If the parameters baseObjectClass and baseObjectInstance are provided the object identified by them has to exist. If they are not provided this pre-condition is not applicable.

~~There is no pre-condition.~~

6.3.2.5 Post-condition

allAlarmInformationReturned.

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

6.3.2.6 Exceptions

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

End of Change in Clause 6.3.2
End of Document

Annex D (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
Mar 2000	S_07	SP-000012	--	--	Approved at TSG SA #7 and placed under Change Control	2.0.0	3.0.0
Mar 2000	--	--	--	--	Cosmetic	3.0.0	3.0.1
Jun 2000	S_08	SP-000250	004	--	Split of TS - Part 2: Alarm Integration Reference Point (IRP): Information Service (IS)	3.0.1	3.1.0
Sep 2000	--	--	--	--	Cosmetic	3.1.0	3.1.1
Sep 2000	S_09	SP-000438	001	--	Correction of qualifier for SystemDN	3.1.1	3.2.0
Sep 2000	S_09	SP-000438	002	--	Addition of a missing constraint in acknowledgeAlarm operation	3.1.1	3.2.0
Dec 2000	S_10	SP-000520	003	--	Incorrect modifiable attributes	3.2.0	3.3.0
Dec 2000	S_10	SP-000520	004	--	Add acknowledgement information to getAlarmList result	3.2.0	3.3.0
Dec 2000	S_10	SP-000520	005	--	Identification of valid Event Types and Extended Event Types within Notifications	3.2.0	3.3.0
Dec 2000	S_10	SP-000520	006	--	A cleared Alarm shall be given perceived severity "Cleared" and nothing else	3.2.0	3.3.0
Dec 2000	S_10	SP-000520	007	--	Inconsistent behaviour for cleared not yet acknowledged alarms	3.2.0	3.3.0
Jun 2001	S_12	SP-010282	008	--	Alarm IRP: IS Rel4 - Addition of feature	3.3.1	4.0.0
Sep 2001	S_13	SP-010474	009	--	Definition of thresholdInfo in Alarm IRP: IS	4.0.0	4.1.0
Dec 2001	S_14	SP-010639	010	--	Correction of notifyChangedAlarm example #2	4.1.0	4.2.0
Dec 2001	S_14	SP-010639	011	--	Update of notificationId missing in To-state of notifyClearedAlarm	4.1.0	4.2.0
Mar 2002	S_15	SP-020028	012	--	Addition of "perceivedSeverity" as parameter to "acknowledgeAlarms operation" (IS)	4.2.0	4.3.0
Mar 2002	S_15	SP-020039	013	--	Addition of parameter in Alarm List Rebuilt notification	4.2.0	4.3.0
Mar 2002	S_15	SP-020039	014	--	Addition of new notification notifyPotentialFaultyAlarmList	4.2.0	4.3.0
Mar 2002	S_15	SP-020039	015	--	Additional trigger event for notifyAlarmListRebuilt	4.2.0	4.3.0
Mar 2002	S_15	--	--	--	Automatic upgrade to Rel-5 (no Rel-5 CR)	4.3.0	5.0.0
Sep 2002	S_17	SP-020477	017	--	Add clearAlarms() operation for Alarm IRP:IS	5.0.0	5.1.0
Sep 2002	S_17	SP-020478	018	--	Add security alarms support in Alarm IRP: IS	5.0.0	5.1.0
Dec 2002	S_18	SP-020751	020	--	Add additionalInformation parameter in notification in Alarm IRP: IS	5.1.0	5.2.0
Mar 2003	S_19	SP-030062	022	--	Add Missing ITU-T M.3100 Probable Causes	5.2.0	5.3.0
Mar 2003	S_19	SP-030063	024	--	Corrections regarding Alarm Acknowledgement and Alarm Comments - alignment with 32.111-1	5.2.0	5.3.0
Mar 2003	S_19	SP-030138	025	--	Add Missing security event types and probable causes	5.2.0	5.3.0
Jun 2003	S_20	SP-030275	026	--	Correction of imported references table for managedGenericIRP	5.3.0	5.4.0

CR-Form-v7

CHANGE REQUEST

⌘ 32.111-3 CR 033 ⌘rev - ⌘ Current version: 5.4.0 ⌘

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

Proposed change affects: UICC apps⌘ [ ] ME [ ] Radio Access Network [X] Core Network [X]

Title:	⌘	Align operation *getAlarmList* with the notification *notifyAlarmListRebuilt*
Source:	⌘	**SA5** (olaf.pollakowski@siemens.com)
Work item code:	⌘	OAM-NIM
Date:	⌘	21/11/2003
Category:	⌘	**B**
Release:	⌘	Rel-6

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)*
Rel-6	*(Release 6)*

Reason for change:	⌘	Through IS notification "notifyAlarmListRebuilt" the IRPAgent can indicate to the IRPManager that only part of the alarm list has been rebuilt, but the IRPManager cannot retrieve through IS operation "getAlarmList" solely the part of the alarm list that the IRPAgent has indicated as having been rebuilt.
Summary of change:	⌘	• Addition of CORBA types "DN" and "DNTypeOpt" to CORBA module "AlarmIRPConstDefs" • Addition of CORBA input parameter "base_object" of CORBA type "DNTypeOpt" to CORBA method "get_alarm_list" in CORBA module "AlarmIRPSystem" • Editorial corrections
Consequences if not approved:	⌘	Functionalities provided by IS operation "getAlarmList" and by IS notification "notifyAlarmListRebuilt" would not be aligned.

Clauses affected:	⌘	1, 5.2, A.1, A.2

		Y	N		
Other specs affected:	⌘		X	Other core specifications	⌘
			X	Test specifications	
			X	O&M Specifications	

Other comments:	⌘	The attached CORBA IDL files "AlarmIRPConstDefs.idl" and "AlarmIRPSystem.idl" reflect the changes from this CR (only).

Change in Clause 1

1 Scope

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

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

This Solution Set specification is related to TS 32.111-2 V6~~5~~.0.X.

End of Change in Clause 1

Change in Clause 5.2

5.2 Operation parameter mapping

●●●

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

IS Operation parameter	SS Method parameter	Qualifier
alarmAckState, filter	ManagedGenericIRPConstDefs::StringTypeOpt filter	O
baseObjectClass, baseObjectInstance	AlarmIRPConstDefs::DNTypeOpt base_object	O, see note
alarmInformation List	Return value of type AlarmIRPConstDefs::AlarmInformationSeq	M
status	Exceptions: GetAlarmList, ManagedGenericIRPSystem::ParameterNotSupported, ManagedGenericIRPSystem::InvalidParameter	M
NOTE: If notification `notifyAlarmListRebuilt` supports indicating that only a part of the alarm list has been rebuilt then this parameter shall be supported.		

●●●

End of Change in Clause 5.2

Change in Annex Clause A.1

A.1 IDL specification (file name "AlarmIRPConstDefs.idl")

```
#ifndef AlarmIRPConstDefs_idl
#define AlarmIRPConstDefs_idl

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

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

/* ## Module: AlarmIRPConstDefs
This module contains commonly used definitions for Alarm IRP
```

```
================================================================
*/
module AlarmIRPConstDefs
{
   /*
   The format of Distinguished Name (DN) is specified in 3GPP TS 32.300
   "Name Conventions for Managed Objects".
   */
   typedef string DN;

   /* DNTypeOpt is an optional type.
   If the discriminator is true the value is present.
   Otherwise the value is null.
   */
   union DNTypeOpt 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_M3100 = 3;
      const short FAR_END_RECEIVER_FAILURE = 4;
      const short FRAMING_ERROR_M3100 = 5;
      const short LOSS_OF_FRAME = 6;
      const short LOSS_OF_POINTER = 7;
      const short LOSS_OF_SIGNAL = 8;
      const short PAYLOAD_TYPE_MISMATCH = 9;
      const short TRANSMISSION_ERROR = 10;
      const short REMOTE_ALARM_INTERFACE = 11;
      const short EXCESSIVE_BIT_ERROR_RATE = 12;
      const short PATH_TRACE_MISMATCH = 13;
      const short UNAVAILABLE = 14;
      const short SIGNAL_LABEL_MISMATCH = 15;
      const short LOSS_OF_MULTI_FRAME = 16;
      const short COMMUNICATIONS_RECEIVE_FAILURE = 17;
      const short COMMUNICATIONS_TRANSMIT_FAILURE = 18;
      const short MODULATION_FAILURE = 19;
      const short DEMODULATION_FAILURE = 20;
      // 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_DEVICE_PROBLEM = 54;
      const short LINE_CARD_PROBLEM = 55;
      const short MULTIPLEXER_PROBLEM_M3100 = 56;
      const short NE_IDENTIFIER_DUPLICATION = 57;
      const short POWER_PROBLEM_M3100 = 58;
      const short PROCESSOR_PROBLEM_M3100 = 59;
      const short PROTECTION_PATH_FAILURE = 60;
      const short RECEIVER_FAILURE_M3100 = 61;
      const short REPLACEABLE_UNIT_MISSING = 62;
      const short REPLACEABLE_UNIT_TYPE_MISMATCH = 63;
      const short SYNCHRONISATION_SOURCE_MISMATCH = 64;
      const short TERMINAL_PROBLEM = 65;
      const short TIMING_PROBLEM_M3100 = 66;
      const short TRANSMITTER_FAILURE_M3100 = 67;
      const short TRUNK_CARD_PROBLEM = 68;
      const short REPLACEABLE_UNIT_PROBLEM = 69;
      const short REAL_TIME_CLOCK_FAILURE = 70;
      // 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 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;
// 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_M3100 = 151;
const short MEMORY_MISMATCH = 152;
const short CORRUPT_DATA_M3100 = 153;
const short OUT_OF_CPU_CYCLES = 154;
const short SOFTWARE_ENVIRONMENT_PROBLEM = 155;
const short SOFTWARE_DOWNLOAD_FAILURE = 156;
const short LOSS_OF_REAL_TIME = 157;
const short REINITIALIZED = 158;
// 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_REDUCTION = 303;
// Value 304 corresponds to a duplicated probable cause
const short COMMUNICATION_PROTOCOL_ERROR = 305;
const short COMMUNICATION_SUBSYSTEM_FAILURE = 306;
const short CONFIGURATION_OR_CUSTOMIZING_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_DETECTION = 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 originating from GSM 12.11.
Values below correspond to GSM 12.11 values with an offset of 500.
*/
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;
// 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_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;
// 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_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;
// 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 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;
// Value 575 corresponds to a duplicated probable cause
// Values 576-700 are reserved for potential GSM 12.11 future extensions
/*
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";
   };

   /*
   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};


   /*
   It indicates if the AlarmList alignment is required.
   */
   enum AlarmListAlignmentRequirementType {Required, NotRequired};



   /* 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~~ gauge 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~~ thresholds. 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 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.
      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: perceived_severity is an optional
   parameter. If this value is present, it must have one of the defined values
   of Interface PerceivedSeverity.
   */
   struct AlarmInformationIdAndSev
   {
      string alarm_information_reference;
      ManagedGenericIRPConstDefs::ShortTypeOpt perceived_severity;
   };

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

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

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

   /*
   Define the structure returned when the acknowledge operation fails for a set
   of alarm ids.
   A failure category and a reason are provided in order to indicate why the
   operation failed.
```

```
   */
   struct BadAcknowledgeAlarmInfo
   {
      string alarm_information_reference;
      AcknowledgeFailureCategories failure_category;
      string reason;
   };

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

End of Change in Annex Clause A.1

Change in Annex Clause A.2

A.2 IDL specification (file name “AlarmIRPSystem.idl”)

```
#ifndef AlarmIRPSystem_idl
#define AlarmIRPSystem_idl

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

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

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

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

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

      If FALSE is returned, the IRPAgent will automatically destroy the
      iterator.
      */
```

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

      /*
      This method destroys the iterator.
      */
      void destroy();
   };


   interface AlarmIRP
   {
      /*
      Return the list of all supported Alarm IRP versions.

      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::StringTypeOpt 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::StringTypeOpt 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::StringTypeOpt 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::StringTypeOpt clear_system_id,
       out AlarmIRPConstDefs::BadAlarmInformationIdSeq
          bad_alarm_information_id_list
    )
    raises (ClearAlarms, ManagedGenericIRPSystem::ParameterNotSupported,
            ManagedGenericIRPSystem::InvalidParameter);


    /*
    This method returns Alarm Informations.
    If flag is TRUE, all returned Alarm Informations shall be
    in AlarmInformationSeq that contains 0 or more Alarm Informations.
    Output parameter iter shall be useless.
    If flag is FALSE, no Alarm Informations shall be in AlarmInformationSeq.
    IRPAgent needs to use iter to retrieve them.
    */
    AlarmIRPConstDefs::AlarmInformationSeq get_alarm_list (
       in ManagedGenericIRPConstDefs::StringTypeOpt filter,
       in AlarmIRPConstDefs::DNTypeOpt base_object,
       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 ManagedGenericIRPConstDefs::StringTypeOpt 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
```

End of Change in Annex Clause A.2
End of Document

Annex B (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
Mar 2000	S_07	SP-000012	--	--	Approved at TSG SA #7 and placed under Change Control	2.0.0	3.0.0
Mar 2000	--	--	--	--	cosmetic	3.0.0	3.0.1
Jun 2000	S_08	SP-000253	005	--	Split of TS - Part 3: Alarm Integration Reference Point (IRP): CORBA Solution Set (SS)	3.0.1	3.1.0
Sep 2000	S_09	SP-000439	003	--	Correct push_structured_event of push_structured_events	3.1.0	3.2.0
Sep 2000	S_09	SP-000439	004	--	Remove the use of interface to encapsulate const strings	3.1.0	3.2.0
Dec 2000	S_10	SP-000521	001	1	Allow "Structured Event Filterable Body Fields" to be absent if parameters are not used	3.2.0	3.3.0
Dec 2000	S_10	SP-000521	002	1	Specific behaviour of the Iterator	3.2.0	3.3.0
Dec 2000	S_10	SP-000521	005	--	Inconsistent qualifiers	3.2.0	3.3.0
Mar 2001	S_11	SP-010032	006	--	Missing how "Notify Alarm List Rebuilt" reason attribute is located in Structured Event	3.3.0	3.4.0
Mar 2001	S_11	SP-010032	007	--	Use alarmInformationBody in additionalInformation.ackTime	3.3.0	3.4.0
Jun 2001	S_12	SP-010239	008	--	Probable Cause "Intrusion Detection" is missing	3.4.0	3.5.0
Jun 2001	S_12	SP-010282	009	--	Alarm IRP: CORBA SS Rel4 - Addition of feature.	3.5.1	4.0.0
Sep 2001	S_13	SP-010469	010	--	Correction of BadAlarmInformationIdSeq parameter type	4.0.0	4.1.0
Sep 2001	S_13	SP-010474	011	--	Definition of thresholdInfo in Alarm IRP: CORBA SS	4.0.0	4.1.0
Sep 2001	S_13	SP-010522	012	--	Eliminate guesses on IDL file names in Alarm IRP: CORBA SS	4.0.0	4.1.0
Mar 2002	S_15	SP-020015	014	--	Correction of erroneous and addition of missing mapping tables	4.1.0	4.2.0
Mar 2002	S_15	SP-020028	015	--	Addition of "perceivedSeverity" as parameter to "acknowledgeAlarms" operation (CORBA SS)	4.1.0	4.2.0
Mar 2002	S_15	--	--	--	Automatic upgrade to Rel-5 (no Rel-5 CR)	4.2.0	5.0.0
Sep 2002	S_17	SP-020476	017	--	Addition of "indeterminate" probable cause in IDL definition	5.0.0	5.1.0
Sep 2002	S_17	SP-020477	018	--	Add clearAlarm and other updates	5.0.0	5.1.0
Sep 2002	S_17	SP-020478	021	--	Add security alarms support in Alarm IRP: CORBA SS	5.0.0	5.1.0
Sep 2002	S_17	SP-020479	019	--	Add optional string parameters in CORBA Solution Set	5.0.0	5.1.0
Dec 2002	S_18	SP-020751	023	--	Add additionalInformation parameter in notification in Alarm IRP: CORBA SS (Alignment with Information Service in Rel-5 32111-2)	5.1.0	5.2.0
Dec 2002	S_18	SP-020752	024	--	Add notifyPotentialFaultyAlarmList in Alarm IRP: CORBA SS (Alignment with Information Service in Rel-5 32111-2)	5.1.0	5.2.0
Mar 2003	S_19	SP-030064	026	--	Correction of CORBA ALARM_IRP_VERSION in line with adopted Rel-5 policy	5.2.0	5.3.0
Mar 2003	S_19	SP-030062	028	--	Add missing ITU-T M.3100 Probable Cause values & Correct CORBA IDL errors	5.2.0	5.3.0
Mar 2003	S_19	SP-030138	029	--	Correction of CORBA IDL Optional clearSystemId	5.2.0	5.3.0
Jun 2003	S_20	SP-030276	030	--	Correction of CORBA type definition in struct "AlarmInformationIdAndSev"	5.3.0	5.4.0

CR-Form-v7

CHANGE REQUEST

⌘ 32.111-4 CR 025 ⌘rev - ⌘ Current version: 5.6.0 ⌘

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

Proposed change affects: UICC apps⌘ ☐ ME ☐ Radio Access Network X Core Network X

Title:	⌘	Align operation *getAlarmList* with the notification *notifyAlarmListRebuilt*
Source:	⌘	**SA5** (olaf.pollakowski@siemens.com)
Work item code:	⌘	OAM-NIM
Date:	⌘	21/11/2003
Category:	⌘	**B**
Release:	⌘	Rel-6

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)*
Rel-6	*(Release 6)*

Reason for change:	⌘	The functionality of the operation *getAlarmist* is aligned at IS level with the notification *notifyAlarmListRebuilt*. This CR reflects this change in the CMIP SS.
Summary of change:	⌘	The paramters *objectClass* and *objectInstance* identifying the part of the alarm list to be uploaded by *getAlarmList* are added to the M-ACTION parameter 'GetAlarmListInfo'
Consequences if not approved:	⌘	The CMIP SS of the Alarm IRP is not aligned with the IS.

Clauses affected:	⌘	1, 4.1.6, 4.2.3, 5, 6

	Y	N		
Other specs affected: ⌘		X	Other core specifications	⌘
		X	Test specifications	
		X	O&M Specifications	

Other comments:	⌘	Mirror of

Change in Clause 1

1 Scope

The present document defines the alarm integration reference point for the CMIP solution set. In detail:

- clause 4 contains an introduction to some basic concepts of the CMIP interfaces;
- clause 5 contains the GDMO definitions for the Alarm Management over the CMIP interfaces;
- clause 6 contains the ASN.1 definitions supporting the GDMO definitions provided in clause 5.

This Solution Set specification is related to 3GPP TS 32.111-2 (V65.40.X).

End of Change in Clause 1

Change in Clause 4.1.6

4.1.6 Alignment of alarm conditions over the Itf-N

The IRP Manager is able to trigger the alarm conditions alignment using the Action *getAlarmList*

The following specifies the logical steps of the alignment procedure, by describing a possible implementation. Any other implementation showing the same behaviour on the Itf-N interface is compliant with the present document.

- The Manager sends to the Agent a *getAlarmList* request containing the following information:
  - *alarmAckState*, used to select the alarms from the Agent's alarm list for the current alignment (e.g. all active alarms).
  - *baseObjectClass, baseObjectInstance*, identifies the part of the alarm list to be uploaded.
  - *destination*, identifying the destination to which event reports that have passed the filter conditions are sent.
  - *filter*, this optional parameter defines the conditions an alarm notification shall fulfil in order to be forwarded to the Manager. It applies only for the current alignment request.
- After evaluation of the request, the Agent first generates an *alignmentId* value, which unambiguously identifies this alignment process. This value is used by the Manager to correlate alarm reports to the corresponding alignment requests, in case this Manager issues several alarm alignments in parallel.
  - The Agent creates a temporary Event Forwarding Discriminator (EFD) instance for the purpose of this alarm alignment, using the parameters *destination* and *filter* received in the request. If the *filter* parameter is absent in the alarm synchronisation request, all alarm notifications are forwarded to the Manager through this EFD, taking into account both the filter constraint currently active for the event reporting to the manager having invoked the synchronisation request andthe value of the parameter *alarmAckState*.
    The filter is set by the Agent automatically in order to forward only those alarm notifications containing, at the beginning of the field *additionalText*, either the string "(ALIGNMENT-<alignmentId>)" or the string "(ALIGNMENTEND-<alignmentId>)".
- The Agent sends back a *getAlarmList* response, which contains the *alignmentId* described above and the *status* information, indicating the result of the request. (see the message flow in Figure 1).
- The Agent scans now its alarm list. For every alarm, which matches the criteria defined by the *alarmAckState* parameter, the Agent inserts, at the beginning of the field *additionalText*, the string "(ALIGNMENT-<alignmentId>)". According to ITU-T Recommendation X.734 [6], the Agent forwards these alarm notifications towards all EFDs.
  In the last alarm of the list the Agent inserts the string "(ALIGNMENTEND-<alignmentId>)" to indicate the end of the alarm alignment.

NOTE: These alarm notifications can reach the current Manager only via the temporary EFD created for the current alignment. They are filtered out:

a) By all the EFD instances used for "real-time" alarm reporting, due to the presence of the sub-string "ALIGNMENT" in the field *additionalText* (see 3GPP TS 32.304 [10]).

b) By all temporary EFD instances possibly created for parallel alignments, due to the presence of the unambiguous sub-string "<alignmentId>" in the *additionalText* field.

- After sending the last alarm report (identified by the sub-string "ALIGNMENTEND" in the *additionalText)*, the Agent automatically deletes the temporary EFD instance (see figure 1).

At the end of the alarm conditions alignment the acknowledgement state and the comments assigned to each alarm are implicitly synchronised between the IRPAgent and the IRPManager that has requested the alignment.

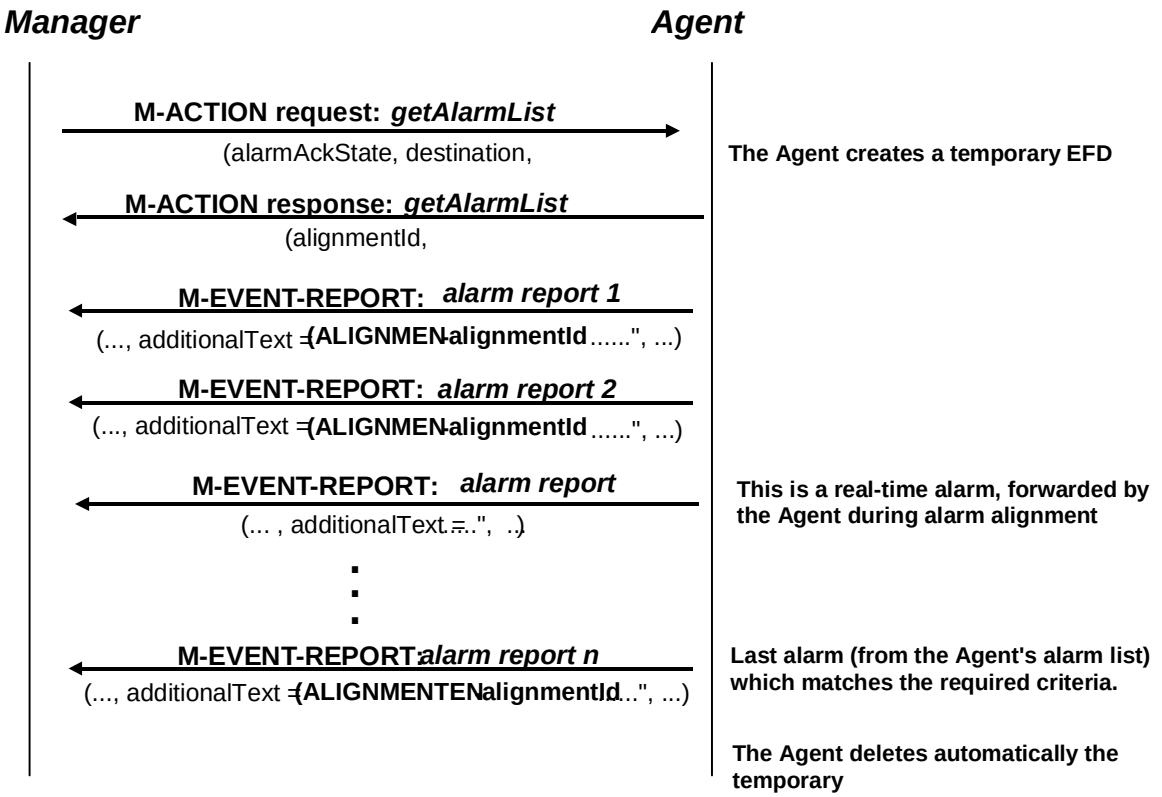

Figure 1: Alignment arrow diagram

Figure 2 shows the handling of a "real-time" alarm notification (occurred during the execution of the *getAlarmList* operation), which is forwarded by the Agent (according to ITU-T Recommendation X.734 [6]) to all currently available EFD instances. Dependent on the *discriminatorConstruct* setting of every EFD, such an alarm may or may not reach the related Manager. In any case, this alarm is filtered out by the temporary EFD assigned to the Manager, which triggered the *getAlarmList* request.

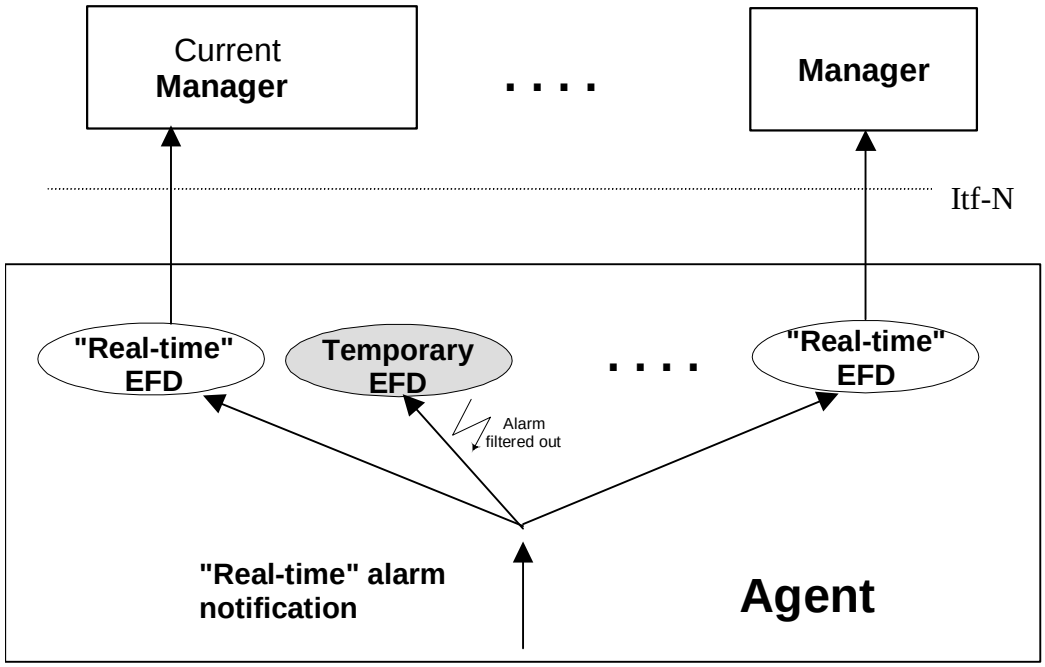

Figure 2: Treatment of "real time" alarms

Figure 3 shows the handling of an alarm notification from the alarm list, matching the criteria defined in the parameters *alarmAckState* of the *getAlarmList* request and forwarded by the Agent to all EFD instances as well. This alarm is filtered out by all EFD instances in charge of discrimination of "real-time" alarms and can reach only the Manager, which triggered the *getAlarmList* request, because it passes the temporary EFD instance assigned to this Manager.

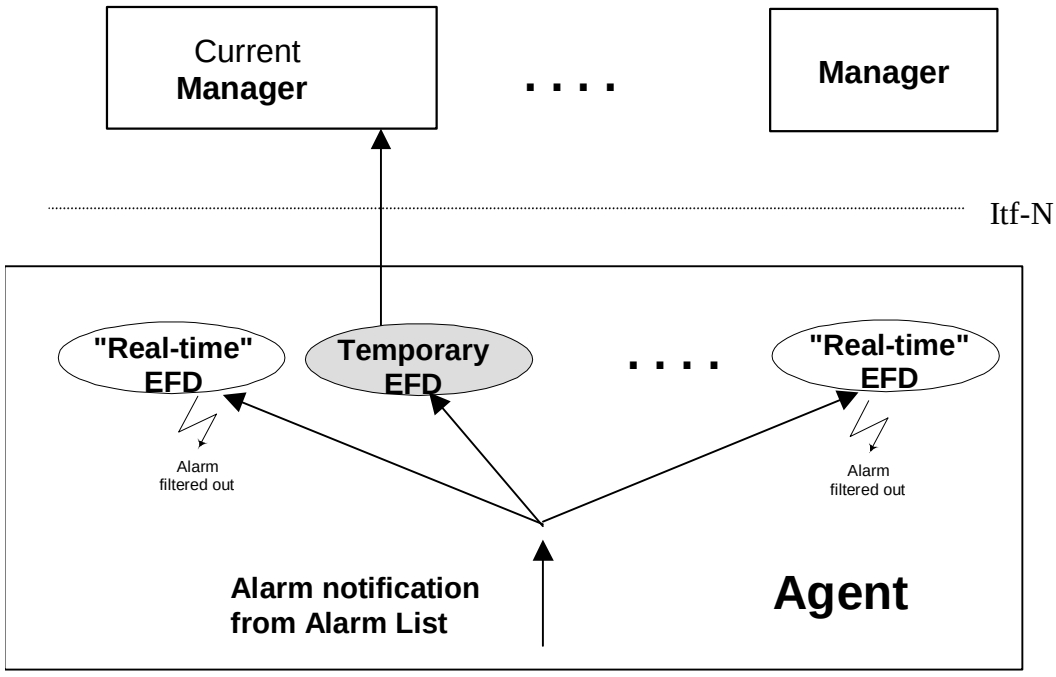

Figure 3: Treatment of "alignment" alarms

End of Change in Clause 4.1.6

Change in Clause 4.2.3

4.2.3 Mapping of Operation Parameters

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: Parameter mapping of the operation *acknowledgeAlarms*

IS Parameter	IN/OUT	CMIP SS Equivalent	Qualifier
alarmInformationAndSeverityReferenceList	IN	M-ACTION parameter 'Action information' (AckOrUnackAlarmsInfo): alarmReferenceList (note)	M
alarmInformationAndSeverityReferenceList	IN	M-ACTION parameter 'Action information' (AckOrUnackAlarmsInfo): AlarmReferenceList (note)	M
ackUserId	IN	M-ACTION parameter 'Action information' (AckOrUnackAlarmsInfo): ackUserId	M
ackSystemId	IN	M-ACTION parameter 'Action information' (AckOrUnackAlarmsInfo): ackSystemId	O
badAlarmInformationReferenceList	OUT	M-ACTION parameter 'Action reply' (AckOrUnackAlarmsReply): errorAlarmReferenceList	M
status	OUT	M-ACTION parameter 'Action reply' (AckOrUnackAlarmsReply): status	M
NOTE: severity verification not required in CMIP solution set.			

Table 3: Parameter mapping of the operation *getAlarmCount*

IS Parameter	IN/OUT	CMIP SS Equivalent	Qualifier
filter	IN	M-ACTION parameter 'Action information' (GetAlarmCountInfo): filter	O
alarmAckState	IN	M-ACTION parameter 'Action information' (GetAlarmCountInfo): alarmAckState	O
criticalCount	OUT	M-ACTION parameter 'Action reply' (GetAlarmCountReply): criticalCount	M
majorCount	OUT	M-ACTION parameter 'Action reply' (GetAlarmCountReply): majorCount	M
minorCount	OUT	M-ACTION parameter 'Action reply' (GetAlarmCountReply): minorCount	M
warningCount	OUT	M-ACTION parameter 'Action reply' (GetAlarmCountReply): warningCount	M
indeterminateCount	OUT	M-ACTION parameter 'Action reply' (GetAlarmCountReply): indeterminateCount	M
clearedCount	OUT	M-ACTION parameter 'Action reply' (GetAlarmCountReply): clearedCount	M
status	OUT	M-ACTION parameter 'Action reply' (GetAlarmCountReply): status	M

Table 4: Parameter mapping of the operation *getAlarmList*

IS Parameter	IN/OUT	CMIP SS Equivalent	Qualifier
filter	IN	filter	O
alarmAckState	IN	alarmAckState	O
baseObjectClass	IN	M-ACTION parameter 'Action information' (GetAlarmListInfo): baseObjectClass	O
baseObjectInstance	IN	M-ACTION parameter 'Action information' (GetAlarmListInfo): baseObjectInstance	O
--		destination (input) - see note 1	M
alarmInformationList	OUT	(sequence of alarm notifications) (see subclause 4.5)	M
status	OUT	status	M
--		alignmentId (output) - see note 2	M
NOTE 1: Destination is a CMIP specific parameter and is determined by the Manager. NOTE 2: AlignmentId is a CMIP specific parameter and is determined by the Agent.			

Table 5: Parameter mapping of the operation *getAlarmIRPVersion*

IS Parameter	IN/OUT	CMIP SS Equivalent	Qualifier
versionNumberSet	OUT	M-ACTION parameter 'Action reply' (GetAlarmIRPVersionReply): versionNumberList	M
status	OUT	M-ACTION parameter 'Action reply' (GetAlarmIRPVersionReply): status	M

Table 6: Parameter mapping of the operation *getOperationProfile*

IS Parameter	IN/OUT	CMIP SS Equivalent	Qualifier
irpVersion	IN	M-ACTION parameter 'Action information': irpVersionNumber	M
operationNameProfile	OUT	M-ACTION parameter 'Action reply' (GetOperationProfileReply): operationNameProfile	M
operationParameterProfile	OUT	M-ACTION parameter 'Action reply' (GetOperationProfileReply): operationParameterProfile	M
status	OUT	M-ACTION parameter 'Action reply' (GetOperationProfileReply): status	M

Table 7: Parameter mapping of the operation *getNotificationProfile*

IS Parameter	IN/OUT	CMIP SS Equivalent	Qualifier
irpVersion	IN	M-ACTION parameter 'Action information': irpVersionNumber	M
notificationNameProfile	OUT	M-ACTION parameter 'Action reply' (GetNotificationProfileReply): notificationNameProfile	M
notificationParameterProfile	OUT	M-ACTION parameter 'Action reply' (GetNotificationProfileReply): notificationParameterProfile	M
status	OUT	M-ACTION parameter 'Action reply' (GetNotificationProfileReply): status	M

Table 8: Parameter mapping of the operation *setComment*

IS Parameter	IN/OUT	CMIP SS Equivalent	Qualifier
alarmInformationReferenceList	IN	M-ACTION parameter 'Action information' (SetCommentInfo): alarmReferenceList	M
commentUserId	IN	M-ACTION parameter 'Action information' (SetCommentInfo): commentUserId	M
commentSystemId	IN	M-ACTION parameter 'Action information' (SetCommentInfo): commentSystemId	O
commentText	IN	M-ACTION parameter 'Action information' (SetCommentInfo): commentText	M
badAlarmInformationReferenceList	OUT	M-ACTION parameter 'Action reply' (SetCommentReply): errorAlarmReferenceList	M
status	OUT	M-ACTION parameter 'Action reply' (SetCommentReply): status	M

Table 9: Parameter mapping of the operation *unacknowledgeAlarms*

IS Parameter	IN/OUT	CMIP SS Equivalent	Qualifier
alarmInformationReferenceList	IN	M-ACTION parameter 'Action information' (AckOrUnackAlarmsInfo): alarmReferenceList	M
ackUserId	IN	M-ACTION parameter 'Action information' (AckOrUnackAlarmsInfo): ackUserId	M
ackSystemId	IN	M-ACTION parameter 'Action information' (AckOrUnackAlarmsInfo): ackSystemId	O
badAlarmInformationReferenceList	OUT	M-ACTION parameter 'Action information' (AckOrUnackAlarmsReply): errorAlarmReferenceList	M
status	OUT	M-ACTION parameter 'Action information' (AckOrUnackAlarmsReply): status	M

Table 10: Parameter mapping of the operation *clearAlarms*

IS Parameter	IN/OUT	CMIP SS Equivalent	Qualifier
alarmInformationReferenceList	IN	M-ACTION parameter 'Action information' (ClearAlarmsInfo): alarmReferenceList	M
clearUserId	IN	M-ACTION parameter 'Action information' (ClearAlarmsInfo): clearUserId	M
clearSystemId	IN	M-ACTION parameter 'Action information' (ClearAlarmsInfo): clearSystemId	O
badAlarmInformationReferenceList	OUT	M-ACTION parameter 'Action reply' (ClearAlarmsReply): errorAlarmReferenceList	M
status	OUT	M-ACTION parameter 'Action reply' (ClearAlarmsReply): status	M

End of Change in Clause 4.2.3

Change in Clause 5 & 6

5 GDMO definitions

5.1 Managed Object Classes

5.1.1 alarmControl

```
alarmControl MANAGED OBJECT CLASS
   DERIVED FROM
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":top;
   CHARACTERIZED BY
```

```
      alarmControlBasicPackage,
      alarmAcknowledgementPackage,
      alarmIRPVersionPackage;
   CONDITIONAL PACKAGES
      alarmCountPackage                      PRESENT IF   "an instance supports it",
      alarmCommentPackage                    PRESENT IF   "an instance supports it",
      alarmProfilePackage                    PRESENT IF   "an instance supports it",
      alarmUnacknowledgementPackage          PRESENT IF   "an instance supports it",
      alarmPotentialFaultyAlarmListPackage   PRESENT IF   "an instance supports it",
      alarmClearPackage                      PRESENT IF   "an instance supports it";
REGISTERED AS {ts32-111AlarmObjectClass 1};
```

5.2 Packages

5.2.1 alarmControlBasicPackage

```
alarmControlBasicPackage PACKAGE
   BEHAVIOUR
      alarmControlBasicPackageBehaviour;
   ATTRIBUTES
      alarmControlId       GET,
      alarmsCountSummary   GET;
   ACTIONS
      getAlarmList;
   NOTIFICATIONS
      notifyAlarmListRebuilt;
REGISTERED AS {ts32-111AlarmPackage 1};

alarmControlBasicPackageBehaviour BEHAVIOUR
DEFINED AS
   "The MOC alarmControl has been defined to provide information to the Manager about the currently
   alarms controlled by the Agent.
   An instance of the 'alarmControl' MOC is identified by the value of the attribute
   'alarmControlId'.
   The attribute 'alarmsCountSummary' provides a summary of the number of alarms managed in the
   Agent's alarm list (including the number of cleared but not yet acknowledged alarms).
   The action 'getAlarmList' is the means, for the Manager, to trigger an alarm alignment procedure
   in accordance with the parameter specified in the action request (this may be needed e.g. for
   first time alignment or after a link interruption between the Agent and the Manager). The alarm
   list is sent as a sequence of single alarm reports.
   The notification 'notifyAlarmListRebuilt' is sent by the Agent to the Manager to inform that the
   alarm list has changed. It is recommended that the Manager subsequently triggers an alarm
   alignment.";
```

5.2.2 alarmCountPackage

```
alarmCountPackage PACKAGE
   BEHAVIOUR
      alarmCountPackageBehaviour;
   ACTIONS
      getAlarmCount;
REGISTERED AS {ts32-111AlarmPackage 2};

alarmCountPackageBehaviour BEHAVIOUR
DEFINED AS
   "This package has been defined to allow the Managers to get information from the Agent about the
   number of alarms currently present in the alarm list.";
```

5.2.3 alarmAcknowledgementPackage

```
alarmAcknowledgementPackage PACKAGE
   BEHAVIOUR
      alarmAcknowledgementPackageBehaviour;
   ACTIONS
      acknowledgeAlarms;
   NOTIFICATIONS
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":communicationsAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":environmentalAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":equipmentAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":processingErrorAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":qualityofServiceAlarm;
REGISTERED AS {ts32-111AlarmPackage 3};
```

```
alarmAcknowledgementPackageBehaviour BEHAVIOUR
DEFINED AS
   "This package has been defined to provide information to the Manager about the acknowledgement
   status of the alarms controlled by the Agent.
   The action 'acknowledgeAlarms' allows the NM operator to acknowledge one or several alarms
   previously sent by the Agent as alarm notifications.
   The ITU-T Recommendation X.721 [4] compliant alarm notifications are sent by the Agent to the
   Manager to inform that one alarm has been acknowledged. The acknowledgement related information
   is carried in the additionalInformation attribute.";
```

5.2.4 alarmUnacknowledgementPackage

```
alarmUnacknowledgementPackage PACKAGE
   BEHAVIOUR
      alarmUnacknowledgementPackageBehaviour;
   ACTIONS
      unacknowledgeAlarms;
   NOTIFICATIONS
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":communicationsAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":environmentalAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":equipmentAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":processingErrorAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":qualityofServiceAlarm;
REGISTERED AS {ts32-111AlarmPackage 4};

alarmUnacknowledgementPackageBehaviour BEHAVIOUR
DEFINED AS
   "This package has been defined to provide the Manager with the capability to un-acknowledge
   alarms.
   The action 'unacknowledgeAlarms' allows the NM operator to un-acknowledge one or several alarms
   previously acknowledged by him.
   The ITU-T Recommendation X.721 [4] compliant alarm notifications are sent by the Agent to the
   Manager to inform that one alarm has been unacknowledged. The acknowledgement related information
   is carried in the additionalInformation attribute.";
```

5.2.5 alarmCommentPackage

```
alarmCommentPackage PACKAGE
   BEHAVIOUR
      alarmCommentPackageBehaviour;
   ACTIONS
      setComment;
   NOTIFICATIONS
      "Rec. X.721 | ISO/IEC 10165-2 : 1992": communicationsAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992": environmentalAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992": equipmentAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992": processingErrorAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992": qualityofServiceAlarm;
REGISTERED AS {ts32-111AlarmPackage 5};

alarmCommentPackageBehaviour BEHAVIOUR
DEFINED AS
   "This package has been defined to allow the management of comments related to alarms.
   The action setComment allows the IRPManager to add a comment to one or several alarms. Also the
   IRPAgent may add comments to alarms.
   ITU-T Recommendation X.721 [4] compliant alarm notifications are generated once a comment is
   added to an alarm. The information in all comments associated to an alarm is carried in the
   attribute additionalInformation.";
```

5.2.6 alarmIRPVersionPackage

```
alarmIRPVersionPackage PACKAGE
   BEHAVIOUR
      alarmIRPVersionPackageBehaviour;
   ATTRIBUTES
      supportedAlarmIRPVersions   GET;
   ACTIONS
      getAlarmIRPVersion;
REGISTERED AS {ts32-111AlarmPackage 6};

alarmIRPVersionPackageBehaviour BEHAVIOUR
DEFINED AS
```

```
   "This package has been defined to allow the Manager to get information about the Alarm IRP
   versions supported by the Agent.
   The attribute 'supportedAlarmIRPVersions' indicates all versions of the Alarm IRP currently
   supported by the Agent.
   The action 'getAlarmIRPVersion' may be invoked by the Manager to get information about the Alarm
   IRP versions supported by the Agent. Such Alarm IRP versions must compatible to each other. This
   means that the Manager may use any one of such Alarm IRP versions";
```

5.2.7 alarmProfilePackage

```
alarmProfilePackage PACKAGE
   BEHAVIOUR
      alarmProfilePackageBehaviour;
   ACTIONS
      getAlarmIRPOperationProfile,
      getAlarmIRPNotificationProfile;
REGISTERED AS {ts32-111AlarmPackage 7};

alarmProfilePackageBehaviour BEHAVIOUR
DEFINED AS
   "This package has been defined to allow the Manager to get detailed information about the profile
   of Alarm IRP.
   The action 'getOperationProfile' is invoked by the Manager to get detailed information about the
   operations supported by Alarm IRP.
   The action 'getNotificationProfile' is invoked by the Manager to get detailed information about
   the notifications supported by Alarm IRP.";
```

5.2.8 alarmPotentialFaultyAlarmListPackage

```
alarmPotentialFaultyAlarmListPackage PACKAGE
   BEHAVIOUR
      alarmPotentialFaultyAlarmListPackageBehaviour;
   NOTIFICATIONS
      notifyPotentialFaultyAlarmList;
REGISTERED AS {ts32-111AlarmPackage 8};

alarmPotentialFaultyAlarmListPackageBehaviour BEHAVIOUR
DEFINED AS
   "This package allows the IRPAgent to inform the IRPManager that the alarm list held by the
   IRPAgent might be faulty.";
```

5.2.9 alarmClearPackage

```
alarmClearPackage PACKAGE
   BEHAVIOUR
      alarmClearPackageBehaviour;
  ACTIONS
      clearAlarms;
REGISTERED AS {ts32-111AlarmPackage 9};

alarmClearPackageBehaviour BEHAVIOUR
DEFINED AS
   "This package allows the IRPManager to clear one or multiple alarms in the IRPAgent.";
```

5.2.10 x721AlarmNotificationsPackage

```
x721AlarmNotificationsPackage PACKAGE
   BEHAVIOUR
      x721AlarmNotificationsPackageBehaviour;
   NOTIFICATIONS
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":communicationsAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":environmentalAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":equipmentAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":processingErrorAlarm,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":qualityofServiceAlarm;
REGISTERED AS {ts32-111AlarmPackage 10};

x721AlarmNotificationsPackageBehaviour BEHAVIOUR
DEFINED AS
   "This package contains all alarm notifications defined in ITU-T X.721.";
```

5.3 Actions

5.3.1 acknowledgeAlarms (M)

```
acknowledgeAlarms ACTION
   BEHAVIOUR
      acknowledgeAlarmsBehaviour;
   MODE
      CONFIRMED;
   WITH INFORMATION SYNTAX
      TS32-111-4TypeModule.AckOrUnackAlarmsInfo;
   WITH REPLY SYNTAX
      TS32-111-4TypeModule.AckOrUnackAlarmsReply;
REGISTERED AS {ts32-111AlarmAction 1};

acknowledgeAlarmsBehaviour BEHAVIOUR
DEFINED AS
   "The behaviour of this functionality is defined within 32.111-2 – below provides an overview and
   CMIP specific semantics.
   This action is invoked by the Manager to indicate to the Agent that one or several alarms
   (previously sent by the Agent as alarm notifications) have to be acknowledged. In the action
   request the NM supplies the parameter ackUserId and ackSystemId. The other acknowledgement
   history parameters, i.e. alarm acknowledgement state (in this case acknowledged) and the
   acknowledgement time are set by the Agent itself.
   The 'Action information' field contains the following data:
      • alarmReferenceList
        This parameter contains a set of MOI (Managed Object Instance) and notificationIdentifier.
        Each pair identifies unambiguously in the scope of the Agent an alarm (previously received
        by the NM) that have to be now acknowledged.  MOI can be absent if scope of uniqueness of
        notificationIdentifier is across the IRPAgent.
      • ackUserId
        It contains the name of the operator who acknowledged the alarm or a generic name
        (dependent on the operational concept). It may have also the value NULL.
      • ackSystemId
        It indicates the management system where the acknowledgment is triggered. It may have also
        the value NULL.
   The 'Action response' contains the following data:
      • status
        This parameter contains the results of the NM acknowledgement action. Possible values:
        noError (0, all alarms found and ack state changed according to the manager request),
        ackPartlySuccessful (some alarms not found / not changeable, see next parameter), error
        (value indicates the reason why the complete operation failed).
      • errorAlarmReferenceList
        This parameter (significant only if status = ackPartlySuccessful) contains the list of moi
        (managed object instance) and notificationIdentifier pairs of the alarms which could not be
        acknowledged and, for each alarm, also the reason of the error.";
```

5.3.2 getAlarmCount (O)

```
getAlarmCount ACTION
   BEHAVIOUR
      getAlarmCountBehaviour;
   MODE
      CONFIRMED;
   WITH INFORMATION SYNTAX
      TS32-111-4TypeModule.GetAlarmCountInfo;
   WITH REPLY SYNTAX
      TS32-111-4TypeModule.GetAlarmCountReply;
REGISTERED AS {ts32-111AlarmAction 2};

getAlarmCountBehaviour BEHAVIOUR
DEFINED AS
   "The behaviour of this functionality is defined within 32.111-2 – below provides an overview and
   CMIP specific semantics.
   The NM invokes this action to receive the number of available alarms in the Agent' alarm list
   according to the specification in the action request. The Manager may use this action to find out
   the number of alarms in the alarm list before invoking a synchronisation by means of the
   getAlarmList operation. The request is possible also before the Manager creates an own event
   forwarding discriminator instance within the Agent.
   The 'Action information' field contains the following data:
      • alarmAckState
        Depending on this optional parameter value, the NM gets the number of alarms of each
        perceivedSeverity value according to the following possible choices:
           -  all alarms
```

- all active alarms (acknowledged or not yet acknowledged)
- all active and acknowledged alarms
- all active and unacknowledged alarms
- all cleared and unacknowledged alarms.

If the parameter is absent, all alarms from the Agent's alarm list are taken into consideration.

- *filter*
  The handling of this optional parameter is as follows:
  - if present and not NULL, it indicates a filter constraint which shall apply in the calculation of the results
  - if its value is NULL, no filter shall be considered and the Agent shall return the number of all alarms according to the value of the parameter *alarmAckState* (see above)
  - if absent, the handling depends on the availability of an event forwarding discriminator instance within the Agent. If this instance is valid, the filter construct of the event forwarding discriminator shall apply. If no EFD instance is available, the Agent shall return the number of all alarms according to the value of the above-mentioned parameter *alarmAckState*.

The 'Action response' is composed of:

- The numbers of alarms for each *perceivedSeverity* value (if applicable).
- The parameter *status* containing the results of the NM action. Possible values: noError (0), error (the value indicates the reason of the error).";

5.3.3 getAlarmList (M)

```
getAlarmList ACTION
   BEHAVIOUR
      getAlarmListBehaviour;
   MODE
      CONFIRMED;
   WITH INFORMATION SYNTAX
      TS32-111-4TypeModule.GetAlarmListInfo;
   WITH REPLY SYNTAX
      TS32-111-4TypeModule.GetAlarmListReply;
REGISTERED AS {ts32-111AlarmAction 3};

getAlarmListBehaviour BEHAVIOUR
DEFINED AS
   "This action starts an alarm alignment procedure between a NM and Agent, which takes into account
   the acknowledgment state of the alarms and a dedicated filter (valid only for the current
   request).
   The 'Action information' field contains the following data:
      • alarmAckState
        Depending on this optional parameter value, the NM gets the alarm reports according to the
        following possible choices:
           - all alarms
           - all active alarms (acknowledged or not yet acknowledged)
           - all active and acknowledged alarms
           - all active and unacknowledged alarms
           - all cleared and unacknowledged alarms.
        If the parameter is absent, all alarms from the Agent's alarm list are taken into
        consideration.
      • baseObjectClass
        This parameter carries the object class of the managed object instance identified by the
        baseObjectInstance parameter.
      • baseObjectInstance
        This parameter carries the DN of a certain managed object instance. Only alarm information
        instances related to this managed object and its subordinate objects shall be provided.
      • destination
        This parameter identifies the destination to which the alarm reports that have passed the
        test conditions specified in the parameter 'filter' are sent. According to ITU-T
        Recommendation X.721 [4], if no destination is specified in the request, then the
        discriminator is created with the destination defaulted to the AE-Title of the invoker.
      • filter
        The handling of this optional parameter (valid only for the current alignment request) is
        as follows:
           - if present and not NULL, it indicates a filter constraint which shall apply in the
             forwarding of the alignment-related alarm reports
           - if its value is NULL, no real filter shall be considered and the Manager receives the
             alarms according to the value of the parameter alarmAckState (see above).
   The 'Action response' contains the following data:
      • alignmentId
        The parameter is defined by the Agent and identifies unambiguously the current alarm
        alignment procedure. It allows the Manager to distinguish between alarm reports sent as
        consequence of several own alignment requests triggered in parallel.
```

- *status*
  The parameter contains the results of the NM action. Possible values: noError (0), error (the value indicates the reason of the error).
  After the action response is forwarded to the NM, the Agent sends the alarm list as a sequence of single alarm notifications in accordance with the values of the request parameters. Every alarm notification contains all fields of the alarm stored in the alarm list. In particular:
- The field *additionalText* contains at the beginning a string to allow a Manager to recognise that this alarm report is sent due to a previous *getAlarmList* request. The structure of this string is:
  - '(ALIGNMENT-alignmentId)' for every alarm report except the last one **or**
  - '(ALIGNMENTEND-alignmentId)' for the last alarm report sent by the Agent due to the current *getAlarmList* request.
- If available, the data related to the acknowledgment history (i.e. ackState, ackTime, ackUserId, ackSystemId) are provided in the field *additionalInformation*.
  Further details about the implementation of this operation are provided in the 'Introduction'.";

5.3.4 setComment (O)

```
setComment ACTION
   BEHAVIOUR
      setCommentBehaviour;
   MODE
      CONFIRMED;
   WITH INFORMATION SYNTAX
      TS32-111-4TypeModule.SetCommentInfo;
   WITH REPLY SYNTAX
      TS32-111-4TypeModule.SetCommentReply;
REGISTERED AS {ts32-111AlarmAction 4};

setCommentBehaviour BEHAVIOUR
DEFINED AS
   "The behaviour of this functionality is defined within 32.111-2 – below provides an overview and
   CMIP specific semantics.
   The NM invokes this action to associate a comment to one or more alarms.
   The 'Action information' field contains:
      • alarmReferenceList
        Contains a list of alarm identifiers to which the comment must be associated.
      • commentUserId
        Contains the identity of the NM User that invokes this operation.
      • commentSystemId
        Contains the identity of the NM that invokes this operation.
      • commentText
        Contains the text of the comment.
   The 'Action response' is composed of the following data:
      • errorAlarmReferenceList
        List of pair of alarmId and failure reason.
      • status
        It contains the results of the NM action. Possible values: actionSucceeded (0),
        actionPartiallyFailed (12) or another value indicating the reason of the error.";
```

5.3.5 getAlarmIRPVersion (M)

```
getAlarmIRPVersion ACTION
   BEHAVIOUR
      getAlarmIRPVersionBehaviour;
   MODE
      CONFIRMED;
   WITH REPLY SYNTAX
      TS32-111-4TypeModule.GetAlarmIRPVersionReply;
REGISTERED AS {ts32-111AlarmAction 5};

getAlarmIRPVersionBehaviour BEHAVIOUR
DEFINED AS
   "The behaviour of this functionality is defined within 32.111-2 – below provides an overview and
   CMIP specific semantics.
   The NM invokes this action to get information about the Alarm IRP versions supported by the
   Agent.
   The 'Action information' field contains no data.
   The 'Action response' is composed of the following data:
      • versionNumbersList
```

It defines a list of Alarm IRP versions supported by the Agent. A list containing no element, i.e. a NULL list means that the concerned Agent doesn't support any version of the Notification IRP.

- *status*
  It contains the results of the NM action. Possible values: noError (0), error (the value indicates the reason of the error).";

5.3.6 getAlarmIRPNotificationProfile (O)

```
getAlarmIRPNotificationProfile ACTION
   BEHAVIOUR
      getAlarmIRPNotificationProfileBehaviour;
   MODE
      CONFIRMED;
   WITH INFORMATION SYNTAX
      TS32-111-4TypeModule.IRPVersionNumber;
   WITH REPLY SYNTAX
      TS32-111-4TypeModule.GetNotificationProfileReply;
REGISTERED AS {ts32-111AlarmAction 6};

getAlarmIRPNotificationProfileBehaviour BEHAVIOUR
DEFINED AS
   "The behaviour of this functionality is defined within 32.111-2 – below provides an overview and
   CMIP specific semantics.
   A Manager invokes this action to enquiry about the notification profile (supported notifications
   and supported parameters) for this specific Alarm IRP version.
   The 'Action information' contains the following data:
      • irpVersionNumber
        This mandatory parameter identifies the Alarm IRP version.
   The 'Action response' is composed of the following data:
      • notificationNameProfile
        It contains a list of notification names, i.e. a NULL list means that the Alarm IRP doesn't
        support any notification.
      • notificationParameterProfile.
        It contains a set of elements, each element corresponds to a notification name and is
        composed by a set of parameter names.
      • status
        It contains the results of this action. Possible values: noError (0), error (the value
        indicates the reason of the error).";
```

5.3.7 getAlarmIRPOperationProfile (O)

```
getAlarmIRPOperationProfile ACTION
   BEHAVIOUR
      getAlarmIRPOperationProfileBehaviour;
   MODE
      CONFIRMED;
   WITH INFORMATION SYNTAX
      TS32-111-4TypeModule.IRPVersionNumber;
   WITH REPLY SYNTAX
      TS32-111-4TypeModule.GetOperationProfileReply;
REGISTERED AS {ts32-111AlarmAction 7};

getAlarmIRPOperationProfileBehaviour BEHAVIOUR
DEFINED AS
   "The behaviour of this functionality is defined within 32.111-2 – below provides an overview and
   CMIP specific semantics.
   A Manager invokes this action to enquiry about the operation profile (supported operations and
   supported parameters) for this specific Alarm IRP version.
   The 'Action information' contains the following data:
      • irpVersionNumber
        This mandatory parameter identifies the Alarm IRP version.
   The 'Action response' is composed of the following data:
      • operationNameProfile
        It contains a list of operation names.
      • operationParameterProfile.
        It contains a set of elements, each element corresponds to an operation name and is
        composed by a set of parameter names.
      • status
        It contains the results of this action. Possible values: noError (0), error (the value
        indicates the reason of the error).";
```

5.3.8 unacknowledgeAlarms (O)

```
unacknowledgeAlarms ACTION
   BEHAVIOUR
      unacknowledgeAlarmsBehaviour;
   MODE
      CONFIRMED;
   WITH INFORMATION SYNTAX
      TS32-111-4TypeModule.AckOrUnackAlarmsInfo;
   WITH REPLY SYNTAX
      TS32-111-4TypeModule.AckOrUnackAlarmsReply;
REGISTERED AS {ts32-111AlarmAction 8};

unacknowledgeAlarmsBehaviour BEHAVIOUR
DEFINED AS
   "The behaviour of this functionality is defined within 32.111-2 – below provides an overview and
   CMIP specific semantics.
   This action is used by the Manager to indicate to the Agent that one or several alarms
   (previously acknowledged) have to be unacknowledged. Subsequently the 'acknowledgement history'
   information of these alarms in the Agent's alarm list is completely removed (this operation may
   be used by operators in case of a previous acknowledgement by mistake).
   The 'Action information' field contains the following data:
      • alarmReferenceList
        This parameter contains a set of MOI (Managed Object Instance) and notificationIdentifier
        pair. Each of them identifies unambiguously in the scope of the Agent an alarm (previously
        acknowledged by the NM) that have to be now unacknowledged. MOI can be absent if scope of
        uniqueness of notificationIdentifier is across the IRPAgent.
      • ackUserId
        It contains the name of the operator who unacknowledged the alarm or a generic name
        (dependent on the operational concept). It may have also the value NULL. Note that only the
        user who previously acknowledged the alarm is allowed to un-acknowledge it later.
      • ackSystemId
        It indicates the management system where the acknowledgment is triggered. It may have also
        the value NULL. Note that the un-acknowledgement is allowed only at the management system
        where previously the acknowledgement took place.
   The 'Action response' contains the following data:
      • status
        This parameter contains the results of the NM un-acknowledgement action. Possible values:
        noError (0, all alarms found and ack state changed according to the manager request),
        unackPartlySuccessful (some alarms not found / not changeable, see next response
        parameter), error (value indicates the reason why the complete operation failed).
      • errorAlarmReferenceList
        This parameter (significant only if status = unackPartlySuccessful) contains the list of
        MOI (Managed Object Instance) and notificationIdentifier pairs of the alarms which could
        not be unacknowledged and, for each alarm, also the reason of the error. MOI can be absent
        if scope of uniqueness of notificationIdentifier is across the IRPAgent. ";
```

5.3.9 clearAlarms (O)

```
clearAlarms ACTION
   BEHAVIOUR
      clearAlarmsBehaviour;
   MODE
      CONFIRMED;
   WITH INFORMATION SYNTAX
      TS32-111-4TypeModule.ClearAlarmsInfo;
   WITH REPLY SYNTAX
      TS32-111-4TypeModule.ClearAlarmsReply;
REGISTERED AS {ts32-111AlarmAction 9};

clearAlarmsBehaviour BEHAVIOUR
DEFINED AS
   "The behaviour of this functionality is defined within 32.111-2 – below provides an overview and
   CMIP specific semantics.
   This action is invoked by the IRPManager to clear manually one or multiple alarms.The M-ACTION
   request parameter 'Action information' ClearAlarmsInfo is composed of the following fields:
      • alarmReferenceList
        This mandatory parameter identifies the alarms to be cleared. Each alarm is identified by
        the notification identifier of the notification that reported the alarm the first time and,
        if the notification identifier is not unique across the IRPAgent, by the instance of the
        managed object that emitted this notification.
      • clearUserId
        This mandatory parameter identifies the user that has invoked the clearAlarms operation.
      • clearSystemId
```

This optional parameter identifies the system on which the IRPManager, where the *clearAlarms* operation has been invoked, is running. This parameter may be absent.

The M-ACTION response parameter 'Action Reply' *ClearAlarmsReply* is composed of the following fields

- *errorAlarmReferenceList*
  This mandatory parameter identifies alarms that are specified in the *alarmReferenceList*, but which could not be cleared. The alarms are specified by the notification identifier of the notification that reported the alarm the first time and, if required, the instance of the managed object that emitted this notification. In addition to this, the parameter specifies for every alarm that could not be cleared the error reason. If all alarms specified in the *alarmReferenceList* exist and could be cleared, this parameter contains no information. If the operation failed completely due to a general error, this parameter is not significant.
- *status*
  This mandatory parameter provides informations about the result of the operation. If all alarms specified in the *alarmReferenceList* exist and are cleared, the value *noError* (0) is returned. If some alarms specified do not exist or could not be cleared, the value *clearPartlySuccessful* () is returned. In this case the parameter *errorAlarmReferenceList* provides additional information. If the operation failed completely due to a general error, this parameter returns the error reason.";

5.4 Notifications

5.4.1 notifyAlarmListRebuilt (M)

```
notifyAlarmListRebuilt NOTIFICATION
   BEHAVIOUR
      alarmListRebuiltBehaviour;
   WITH INFORMATION SYNTAX
      TS32-111-4TypeModule.NotifyAlarmListRebuiltInfo;
REGISTERED AS {ts32-111AlarmNotification 1};

alarmListRebuiltBehaviour BEHAVIOUR
DEFINED AS
   "This notification is used by the Agent to inform the NM that the alarm list has been rebuilt.
   The 'Event Information' field contains the following data:
      • notificationIdentifier
        This ITU-T X.721 standardised parameter, together with MOI (Managed Object Instance),
        unambiguously identifies this notification.
      • rebuiltObjectClass
        This parameter carries the IRPAgent MOC when the entire AlarmList has been rebuilt. It
        carries a different MOC when the AlarmList has been partially rebuilt.
      • rebuiltObjectInstance
        This parameter carries DN of the IRPAgent when the entire AlarmList has been rebuilt. It
        carries the DN of another MOI when the AlarmList has been partially rebuilt  and only the
        MOIs subordinate of  this rebuilt MOI may be affected by this partial rebuilt.
      • reason
        The parameter indicates the reason for alarm list rebuilding (if applicable).
      • alarmListAlignmentRequirement
        This parameter indicates, if the IRPManager has to align its alarm list with the IRPAgent.
        Absence of this parameter means, that an alignment is required. ";
```

5.4.2 notifyPotentialFaultyAlarmList (O)

```
notifyPotentialFaultyAlarmList NOTIFICATION
   BEHAVIOUR
      notifyPotentialFaultyAlarmListBehaviour;
   WITH INFORMATION SYNTAX
      TS32-111-4TypeModule.NotifyPotentialFaultyAlarmListInfo;
REGISTERED AS {ts32-111AlarmNotification 3};

notifyPotentialFaultyAlarmListBehaviour BEHAVIOUR
DEFINED AS
   "This notification is used by the IRPAgent to inform the IRPManager that the IRPAgent has lost
   confidence in the integrity of its alarm list.
   The 'Event information' field contains the following data:
      • potentialFaultyObjectClass
        This parameter specifies together with the parameter potentialFaultyObjectInstance the
        unreliable alarm information instances in the alarm list.
        If this parameter carries the MOC of the IRPAgent, then the entire alarm list is
        unreliable.
```

If this parameter carries the MOC of another MO, then only a part of the alarm list is unreliable. The mechanism for identifying the unreliable part is described below.

- *potentialFaultyObjectInstance*
  This parameter specifies together with the parameter *potentialFaultyObjectClass* the unreliable alarm information instances in the alarm list.
  If *potentialFaultyObjectClass* carries the MOC of the IRPAgent, the this parameter carries the DN of the IRPAgent and the entire alarm list is unreliable.
  If *potentialFaultyObjectClass* carries the MOC of another MO, then this parameter carries the DN of an instance of this class. All alarm information instances representing alarms raised by this MOI and its subordinates may be unreliable in this case.
- *notificationIdentifier*
  This parameter specifies the notification identifier (ITU-T X.733 [5]), which, together with the instance of the object emitting this notification, unambiguously identifies this notification.
- *reason*
  This parameter specifies the reason why the IRPAgent has lost confidence in the integrity of its alarm list and needs to rebuild it.";

5.5 Attributes

5.5.1 alarmControlId

```
alarmControlId ATTRIBUTE
   WITH ATTRIBUTE SYNTAX
      TS32-111-4TypeModule.GeneralObjectId;
   MATCHES FOR
      EQUALITY;
   BEHAVIOUR
      alarmControlIdBehaviour;
REGISTERED AS {ts32-111AlarmAttribute 1};

alarmControlIdBehaviour BEHAVIOUR
DEFINED AS
   "This attribute names an instance of a 'alarmControl' object class.";
```

5.5.2 alarmsCountSummary

```
alarmsCountSummary ATTRIBUTE
   WITH ATTRIBUTE SYNTAX
      TS32-111-4TypeModule.AlarmsCountSummary;
   MATCHES FOR
      EQUALITY;
   BEHAVIOUR
      alarmsCountSummaryBehaviour;
REGISTERED AS {ts32-111AlarmAttribute 2};

alarmsCountSummaryBehaviour BEHAVIOUR
DEFINED AS
   "This attribute indicates a summary of number of alarms managed in the Agent's alarm list sorted
   according to the perceived severity (including the number of cleared but not yet acknowledged
   alarms). Additionally the number of all currently active alarms is provided.";
```

5.5.3 supportedAlarmIRPVersions

```
supportedAlarmIRPVersions ATTRIBUTE
   WITH ATTRIBUTE SYNTAX
      TS32-111-4TypeModule.SupportedAlarmIRPVersions;
   MATCHES FOR
      EQUALITY;
   BEHAVIOUR
      supportedAlarmIRPVersionsBehaviour;
REGISTERED AS {ts32-111AlarmAttribute 3};

supportedAlarmIRPVersionsBehaviour BEHAVIOUR
DEFINED AS
   "This attribute provides the information concerning the Alarm IRP versions currently supported by
   the Agent.";
```

5.6 Parameters

5.6.1 ackStateParameter

```
ackStateParameter PARAMETER
   CONTEXT
      TS32-111-4TypeModule.AlarmInfo.additionalInformation;
   WITH SYNTAX
      TS32-111-4TypeModule.AckState;
   BEHAVIOUR
      ackStateParameterBehaviour;
REGISTERED AS {ts32-111AlarmParameter 1};

ackStateParameterBehaviour BEHAVIOUR
DEFINED AS
   "This parameter models the optional additionalInformation field of the alarm notification. If
   present, it informs the NM about the current acknowledgement state of the present alarm.";
```

5.6.2 ackSystemIdParameter

```
ackSystemIdParameter PARAMETER
   CONTEXT
      TS32-111-4TypeModule.AlarmInfo.additionalInformation;
   WITH SYNTAX
      TS32-111-4TypeModule.SystemId;
   BEHAVIOUR
      ackSystemIdParameterBehaviour;
REGISTERED AS {ts32-111AlarmParameter 2};

ackSystemIdParameterBehaviour BEHAVIOUR
DEFINED AS
   "This parameter models the optional additionalInformation field of the alarm notification. If
   present, it informs the NM about the identifier of the management system where the present alarm
   has been acknowledged.";
```

5.6.3 ackTimeParameter

```
ackTimeParameter PARAMETER
   CONTEXT
      TS32-111-4TypeModule.AlarmInfo.additionalInformation;
   WITH SYNTAX
      TS32-111-4TypeModule.AckTime;
   BEHAVIOUR
      ackTimeParameterBehaviour;
REGISTERED AS {ts32-111AlarmParameter 3};

ackTimeParameterBehaviour BEHAVIOUR
DEFINED AS
   "This parameter models the optional additionalInformation field of the alarm notification. If
   present, it informs the NM about the time the present alarm has been acknowledged by the Agent.";
```

5.6.4 ackUserIdParameter

```
ackUserIdParameter PARAMETER
   CONTEXT
      TS32-111-4TypeModule .AlarmInfo.additionalInformation;
   WITH SYNTAX
      TS32-111-4TypeModule.UserId;
   BEHAVIOUR
      ackUserIdParameterBehaviour;
REGISTERED AS {ts32-111AlarmParameter 4};

ackUserIdParameterBehaviour BEHAVIOUR
DEFINED AS
   "This parameter models the optional additionalInformation field of the alarm notification. If
   present, it informs the NM about the identifier of the user who acknowledged the present alarm.";
```

5.6.5 clearUserIdParameter

```
clearUserIdParameter PARAMETER
```

```
   CONTEXT
      TS32-111-4TypeModule .AlarmInfo.additionalInformation;
   WITH SYNTAX
      TS32-111-4TypeModule.UserId;
   BEHAVIOUR
      clearUserIdParameterBehaviour;
REGISTERED AS {ts32-111AlarmParameter 5};

clearUserIdParameterBehaviour BEHAVIOUR
DEFINED AS
   "This parameter is carried by additionalInformation in the notification reporting the clearance
   of an alarm. It identifies the user that has invoked the clearAlarms operation, that has led to
   the clearance of the reported alarm clearance.";
```

5.6.6 clearSystemIdParameter

```
clearSystemIdParameter PARAMETER
   CONTEXT
      TS32-111-4TypeModule.AlarmInfo.additionalInformation;
   WITH SYNTAX
      TS32-111-4TypeModule.UserId;
   BEHAVIOUR
      clearSystemIdParameterBehaviour;
REGISTERED AS {ts32-111AlarmParameter 6};

clearSystemIdParameterBehaviour BEHAVIOUR
DEFINED AS
   "This parameter is carried by additionalInformation in the notification reporting the clearance
   of an alarm. It identifies the system on which the IRPManager, where the clearAlarms operation
   that has led to the clearance of the reported alarm, is running";
```

5.6.7 commentsParameter

```
commentsParameter PARAMETER
   CONTEXT
      TS32-111-4TypeModule.AlarmInfo.additionalInformation;
   WITH SYNTAX
      TS32-111-4TypeModule.AlarmComments;
   BEHAVIOUR
      commentsParameterBehaviour;
REGISTERED AS   {ts32-111AlarmParameter 7};

commentsParameterBehaviour BEHAVIOUR
DEFINED AS
   "This parameter is carried by the attribute additionalInformation in alarm notifications. If
   present, it informs the IRPManager about the comments assigned to an alarm. Every single comment
   includes the following data: commentText, commentTime, commentUserId and (optionally)
   commentSystemId.";
```

6 ASN.1 definitions for Alarm IRP

```
TS32-111-4TypeModule {itu-t(0) identified-organization(4) etsi(0) mobileDomain(0) umts-Operation-
Maintenance(3) ts-32-111(111) part4(4) informationModel(0) asn1Module(2) version1(1)}

DEFINITIONS IMPLICIT TAGS ::=
BEGIN

--EXPORTS everything

IMPORTS

NotificationIdentifier, Destination, EventTime, ProbableCause, PerceivedSeverity
   FROM Attribute-ASN1Module {joint-iso-ccitt ms(9) smi(3) part2(2) asn1Module(2) 1}

AlarmInfo
   FROM Notification-ASN1Module {joint-iso-ccitt ms(9) smi(3) part2(2) asn1Module(2) 2}

CMISFilter, ObjectInstance, ObjectClass, EventTypeId
   FROM CMIP-1 {joint-iso-ccitt ms(9) cmip(1) modules(0) protocol(3)};

baseNodeUMTS                 OBJECT IDENTIFIER ::= {itu-t (0) identified-organization (4)
                                                    etsi (0) mobileDomain (0)
                                                    umts-Operation-Maintenance (3)}

ts32-111Prefix               OBJECT IDENTIFIER ::= {baseNodeUMTS ts-32-111(111)}
ts32-111Part4                OBJECT IDENTIFIER ::= {ts32-111Prefix part4(4)}
ts32-111-4InfoModel          OBJECT IDENTIFIER ::= {ts32-111Part4 informationModel(0)}

ts32-111AlarmObjectClass     OBJECT IDENTIFIER ::= {ts32-111-4InfoModel managedObjectClass(3)}
ts32-111AlarmPackage         OBJECT IDENTIFIER ::= {ts32-111-4InfoModel package(4)}
ts32-111AlarmParameter       OBJECT IDENTIFIER ::= {ts32-111-4InfoModel parameter(5)}
ts32-111AlarmAttribute       OBJECT IDENTIFIER ::= {ts32-111-4InfoModel attribute(7)}
ts32-111AlarmAction          OBJECT IDENTIFIER ::= {ts32-111-4InfoModel action(9)}
ts32-111AlarmNotification    OBJECT IDENTIFIER ::= {ts32-111-4InfoModel notification(10)}



-- Start of 3GPP SA5 own definitions

AckErrorList ::= SET OF ErrorInfo

AlarmReference ::= SEQUENCE
   {
   moi                          ObjectInstance OPTIONAL, -- absent if scope of uniquness of
                                                         -- notificationId is across IRPAgent
   notificationIdentifier       NotificationIdentifier
   }

AckOrUnackAlarmsInfo ::= SEQUENCE
   {
   alarmReferenceList           SET OF AlarmReference,
   ackUserId                    UserId,
   ackSystemId                  SystemId OPTIONAL
   }

AckOrUnackAlarmsReply ::= SEQUENCE
   {
   status                       ErrorCauses,
   errorAlarmReferenceList      AckErrorList
   }

AckState ::= ENUMERATED
   {
   acknowledged    (0),
   unacknowledged  (1)
   }

AckTime ::= GeneralizedTime

AlarmChoice ::= ENUMERATED
   {
   allAlarms                    (0),
   allActiveAlarms              (1),
```

```
   allActiveAndAckAlarms      (2),
   allActiveAndUnackAlarms    (3),
   allClearedAndUnackAlarms   (4),
   allUnackAlarms             (5)
   }

AlarmComments ::= SET OF SingleAlarmComment

AlarmsCountSummary ::= SEQUENCE
   {
   activeAlarmsCount       INTEGER,   -- this is the sum of criticalCount, majorCount,
                                      -- minorCount, warningCount and indeterminateCount
   criticalCount           INTEGER,
   majorCount              INTEGER,
   minorCount              INTEGER,
   warningCount            INTEGER,
   indeterminateCount      INTEGER,
   clearedCount            INTEGER
   }

AlarmListAlignmentRequirement ::= ENUMERATED
   {
   alignmentRequired      (0),   -- An alarm alignment is required.
   alignmentNotRequired   (1)    -- An alarm alignment is not required.
   }

ClearAlarmsInfo ::= SEQUENCE
   {
   alarmReferenceList      SET OF AlarmReference,
   clearUserId             UserId,
   clearSystemId           SystemId OPTIONAL
   }

ClearAlarmsReply ::= SEQUENCE
   {
   status                        ErrorCauses,
   errorAlarmReferenceList       ClearErrorList
   }

ClearErrorList ::= SET OF ErrorInfo

CommentText ::= GraphicString

CommentTime ::= GeneralizedTime

ErrorCauses ::= ENUMERATED
   {
   noError                          (0), -- operation / notification successfully performed
   wrongFilter                      (1), -- the value of the filter parameter is not valid
   wrongAlarmAckState               (2), -- the value of the alarmAckState parameter (e.g.
                                         -- getAlarmCount) is not valid
   ackPartlySuccessful              (3), -- acknowledgment request partly successful
   unackPartlySuccessful            (4), -- unacknowledgment request partly successful
   wrongAlarmReference              (5), -- alarm identifier used in the alarm reference list not
                                         -- found (e.g. in case of acknowledgement request)
   wrongAlarmReferenceList          (6), -- the alarm reference list (e.g. in case of
                                         -- acknowledgement request) is empty or completely wrong
   alarmAlreadyAck                  (7), -- alarm to be acknowledged is already in this state
   alarmAlreadyUnack                (8), -- alarm to be acknowledged is already in this state
   wrongUserId                      (9), -- the user identifier in the unacknowledgement operation
                                         -- is not the same as in the previous
                                         -- acknowledgementAlarms request
   wrongSystemId                   (10), -- the system identifier in the unacknowledgement
                                         -- operation is not the same as in the previous
                                         -- acknowledgementAlarms request
   alarmAckNotAllowed              (11), -- current management system not allowed to acknowledge the
                                         -- alarm (e.g. due to acknowledgement competence rules)
   setCommentPartlySuccessful      (12), -- the setComment action partly successful (e.g. some
                                         -- alarmId are not in the alarmList)
   clearAlarmsPartlySuccessful     (13), -- only some alarms to be cleared could be cleared
   clearAlarmsNotAllowed           (14), -- current management system not allowed to clear the alarm
   clearAlarmsAlarmAlreadyCleared  (15), -- alarm to be cleared is already cleared
   unspecifiedErrorReason         (255)  -- operation failed, specific error unknown
   }

ErrorInfo ::= SEQUENCE
   {
   moi                         ObjectInstance OPTIONAL,   -- absent if uniqueness of
```

```
                                                        -- notificationIdentifier is across
                                                        -- IRPAgent
   notificationIdentifier      NotificationIdentifier,    -- ITU-T X.721
   reason                      ErrorCauses
   }

GeneralObjectId ::= INTEGER

GetAlarmCountInfo ::= SEQUENCE
   {
   alarmAckState      AlarmChoice OPTIONAL,
   filter             CMISFilter OPTIONAL     -- ITU-T X.711
   }

GetAlarmCountReply ::= SEQUENCE
   {
   criticalCount           INTEGER,
   majorCount              INTEGER,
   minorCount              INTEGER,
   warningCount            INTEGER,
   indeterminateCount      INTEGER,
   clearedCount            INTEGER,
   status                  ErrorCauses
   }

GetAlarmIRPVersionReply ::= SEQUENCE
   {
   versionNumberList      SupportedAlarmIRPVersions,
   status                 ErrorCauses
  }

GetAlarmListInfo ::= SEQUENCE
   {
   alarmAckState          AlarmChoice OPTIONAL,
   baseObjectClass        ObjectClass OPTIONAL      -- ITU-T X.711
   baseObjectInstance     ObjectInstance OPTIONAL   -- ITU-T X.711
   destination            Destination,              -- ITU-T X.721
   filter                 CMISFilter OPTIONAL       -- ITU-T X.711
   }

GetAlarmListReply ::= SEQUENCE
   {
   alignmentId      INTEGER,
   status           ErrorCauses
   }

GetNotificationProfileReply ::= SEQUENCE
   {
   notificationNameProfile           NotificationList,
   notificationParameterProfile      ParameterListOfList,
   status                            ErrorCauses
   }

GetOperationProfileReply ::= SEQUENCE
   {
   operationNameProfile           OperationList,
   operationParameterProfile      ParameterListOfList,
   status                         ErrorCauses
   }

IRPVersionNumber ::= GraphicString

NotificationList ::= SET OF NotificationName

NotificationName ::= GraphicString

NotifyAlarmListRebuiltInfo ::= SEQUENCE
   {
   notificationIdentifier             NotificationIdentifier,         -- ITU-T X.721
   rebuiltObjectClass                 ObjectClass,                    -- ITU-T X.721
   rebuiltObjectInstance              ObjectInstance,                 -- ITU-T X.721
   reason                             ReasonAlarmListRebuilt,
   alarmListAlignmentRequirement      AlarmListAlignmentRequirement OPTIONAL
   }

NotifyPotentialFaultyAlarmListInfo ::= SEQUENCE
   {
   potentialFaultyObjectClass         ObjectClass,                     -- ITU-T X.711
```

```
   potentialFaultyObjectInstance      ObjectInstance,                  -- ITU-T X.711
   notificationIdentifier             NotificationIdentifier,          -- ITU-T X.721
   reason                             ReasonPotentialFaultyAlarmList
   }

OperationList ::= SET OF OperationName

OperationName ::= GraphicString

ParameterList ::= SET OF ParameterName

ParameterListOfList ::= SET OF ParameterList

ParameterName ::= GraphicString

ReasonAlarmListRebuilt :: ENUMERATED
   {
   AgentNetworkEntityCommunicationError   (0),
   AgentRestart                           (1),
   Indeterminate                          (2)
   }

ReasonPotentialFaultyAlarmList ::= ENUMERATED
   {
   communicationErrorNEAgent   (0), -- A communication error between a NE and the agent has occured.
   agentRestart                (1), -- The agent has restarted and not yet updated its alarm list.
   indeterminate               (2)  -- The reasn could not be determined.
   }

SetCommentInfo ::= SEQUENCE
   {
   alarmReferenceList      SET OF AlarmReference,
   commentUserId           UserId,
   commentSystemId         [2] SystemId OPTIONAL,
   commentText             CommentText
   }

SetCommentReply ::= SEQUENCE
   {
   errorAlarmReferenceList      SET OF ErrorInfo,
   status                       ErrorCauses
   }

SingleAlarmComment ::= SEQUENCE
   {
   commentText          CommentText,
   commentTime          CommentTime,
   commentUserId        UserId,
   commentSystemId      SystemId OPTIONAL
   }

SystemId ::= GraphicString

SupportedAlarmIRPVersions ::= SET OF IRPVersionNumber

UserId ::= GraphicString

END -- of module TS32-111-4TypeModule
```

End of Change in Clause 5 & 6
End of Document

Annex A (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
Mar 2000	S_07	SP-000012	--	--	Approved at TSG SA #7 and placed under Change Control	2.0.0	3.0.0
Mar 2000	--	--	--	--	cosmetic	3.0.0	3.0.1
Jun 2000	S_08	SP-000254	005	--	Split of TS - Part 4: Alarm Integration Reference Point (IRP): CMIP Solution Set (SS)	3.0.1	3.1.0
Sep 2000	--	--	--	--	cosmetic	3.1.0	3.1.1
Jun 2001	S_12	SP-010282	001	--	Alarm IRP: CMIP SS Rel4 - Addition of feature. As SA5 had not reviewed this part, it is submitted to SA#12 for Information only.	3.1.1	--
Sep 2001	S_13	SP-010470	001	1	Addition of features	3.1.1	4.0.0
Dec 2001	S_14	SP-010640	003	--	Change of qualifier for setComment and notifyComment	4.0.0	4.1.0
Dec 2001	S_14	SP-010640	004	--	Addition of missing parameter in notifyComments	4.0.0	4.1.0
Mar 2002	S_15	SP-020028	005	--	Addition of "perceivedSeverity" as parameter to "acknowledgeAlarms" operation (CMIP SS)	4.1.0	4.2.0
Mar 2002	S_15	--	--	--	Automatic upgrade to Rel-5 (no Rel-5 CR)	4.2.0	5.0.0
Jun 2002	S_16	SP-020283	007	--	Correction of errors and ambiguities in the Parameter Mapping Tables and ASN.1 Definitions	5.0.0	5.1.0
Jun 2002	S_16	SP-020284	008	--	Addition of the parameter alarmListAlignmentRequirement to the notification notifyAlarmListRebuilt in the CMIP SS (32.111-4)	5.0.0	5.1.0
Jun 2002	S_16	SP-020284	009	--	Adding the notification notifyPotentialFaultyAlarmList in the CMIP SS (32.111-4)	5.0.0	5.1.0
Jun 2002	S_16	SP-020284	010	--	Introduction of SS (32.111-4) to IS (32.111-2) relation and correction of Foreword	5.0.0	5.1.0
Sep 2002	S_17	SP-020480	011	--	Alignment with 32.111-2 on Alarm Clearance Functionality	5.1.0	5.2.0
Dec 2002	S_18	SP-020751	013	--	Add the additionalInformation parameter in notifyNewAlarms to the Alarm IRP CMIP SS (Alignment with Information Service in Rel-5 32111-2)	5.2.0	5.3.0
Dec 2002	S_18	SP-020753	014	--	Addition of Security Alarm Support to the Alarm IRP CMIP SS (Alignment with Information Service in Rel-5 32111-2)	5.2.0	5.3.0
Mar 2003	S_19	SP-030063	016	--	Correction to Alarm Comments- alignment with 32.111-1	5.3.0	5.4.0
Mar 2003	S_19	SP-030138	017	--	Add missing x721AlarmNotificationsPackage	5.3.0	5.4.0
Mar 2003	S_19	SP-030138	018	--	Corrections to GDMO and ASN.1 definitions in the Alarm IRP CMIP SS	5.3.0	5.4.0
Jun 2003	S_20	SP-030277	019	--	Correction of Compilation Errors	5.4.0	5.5.0
Jun 2003	S_20	SP-030277	020	--	Addition of missing reasons for the emission of notifyAlarmListRebuilt	5.4.0	5.5.0
Sep 2003	S_21	SP-030416	022	--	Correction of syntax error in type SetCommentInfo	5.5.0	5.6.0