Technical Specification Group Services and System Aspects
Meeting #14, Kyoto, Japan, 17-20 December 2001

TSGS#14(01)0653

Source: **SA5 (Telecom Management)**

Title: **Rel-5 CR 32.304, Rel-5 CR 32.302 (S5-010772, S5-010773)**

Document for: **Decision**

Agenda Item: **7.5.3**

If the following "parent" CR is approved then also the attached "child" CR can be approved: Rel-5 CR 32.302 (S5-010773).

Doc-1st-	Spec	CR	R	Phas	Subject	C	Versi	Versi	Doc-2nd-	Workitem
SP-010653	32.304	004		Rel-5	**Maximise the reuse of ITU-T CMIP event report management functions**	C	4.1.0	5.0.0	S5-010772	OAM-NIM

The below "child" CR can only be approved if the above "parent" Rel-5 CR 32.304 (S5-010772) has been approved.

Doc-1st-	Spec	CR	R	Phas	Subject	C	Versi	Versi	Doc-2nd-	Workitem
SP-010653	32.302	002		Rel-5	**Change from Mandatory to Conditional the qualifier of the output parameter ‘NotificationCategorySet’ of the operation ‘getSubscriptionStatus’**	C	4.1.0	5.0.0	S5-010773	OAM-NIM

3GPP TSG-SA5 (Telecom Management) **S5-010773**
Meeting #24, Cancun, MEXICO, 26 – 30 November 2001 **S5C010516rev3**

CR-Form-v4

CHANGE REQUEST

⌘	**32.302**	**CR**	**002**	⌘ ev	**-**	⌘	Current version:	**4.0.0** ⌘

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

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

Title:	⌘	Change from Mandatory to Conditional the qualifier of the output parameter 'NotificationCategorySet' of the operation 'getSubscriptionStatus'	
Source:	⌘	SA5	
Work item code:	⌘	OAM-NIM	***Date:*** ⌘ 30/11/2001
Category:	⌘	**C**	***Release:*** ⌘ REL-5
		Use one of the following categories: ***F*** *(correction)* ***A*** *(corresponds to a correction in an earlier release)* ***B*** *(addition of feature),* ***C*** *(functional modification of feature)* ***D*** *(editorial modification)* Detailed explanations of the above categories can be found in 3GPP TR 21.900.	*Use one of the following releases:* *2 (GSM Phase 2)* *R96 (Release 1996)* *R97 (Release 1997)* *R98 (Release 1998)* *R99 (Release 1999)* *REL-4 (Release 4)* *REL-5 (Release 5)*

Reason for change:	⌘	The Notification IRP IS (32.302) requires the CMIP SS (32.304) to support the notification category set parameter, but with the CR in S5-010772 implemented, due to the construction of the EFD, it would not.
Summary of change:	⌘	The qualifier of the output parameter 'notificationCategorySet' of the operation 'getSubscriptionStatus' is changed from mandatory to conditional.
Consequences if not approved:	⌘	The CMIP SS would be inconsistent with the IS.

Clauses affected:	⌘	6.5.1.3		
Other specs affected:	⌘	☐ Other core specifications	⌘	
		☐ Test specifications		
		[X] O&M Specifications		32.304
Other comments:	⌘	This "child" CR can only be approved if its "parent" Rel-5 CR 32.304 (S5-010772) has been approved.		

6.5.1 Operation `getSubscriptionStatus` (O)

6.5.1.1 Definition

IRPManager invokes this operation to query the subscription status of a particular subscription. IRPManager can use getSubscriptionStatus operation to know about the filter constraint in effect, the state of subscription (i.e., if subscription is suspended/inactive or resumed/active), the timeTick value that may be set at subscribe invocation time and the notificationCategory currently in used in the subscription.

6.5.1.2 Input parameters

Parameter Name	Qualifier	Information Type	Comment
`subscriptionId`	M	NtfSubscription.ntfSubscriptionId	It holds the `subscriptionId` carried as the output parameter in the `subscribe` operation.

6.5.1.3 Output parameters

Parameter Name	Qualifier	Matching Information	Comment
`notification CategorySet`	~~M~~C	NtfSubscription.ntfNotificationCategorySet	It identifies the notification Category(ies) supported in this subscription.
`filterInEffect`	O	NtfSubscription.ntfFilter	It contains the filter constraint currently set.
`SubscriptionState`	O	NtfSubscription.ntfSubscriptionState	
`timeTick`	O	NtfSubscription.ntfTimeTick	It carries the same value as the one in `subscribe` operation
`status`	M	ENUM (Operation succeeded, Operation failed)	If (timeTickReset) is true, status = OperationSucceeded. If operation_failed is true, status = OperationFailed.

3GPP TSG-SA5 (Telecom Management) **S5-010772**
Meeting #24, Cancun, MEXICO, 26 – 30 November 2001 **S5C010515rev3**

CR-Form-v4

CHANGE REQUEST

⌘	**32.304**	**CR**	**004**	⌘	ev	**-**	⌘	Current version:	**4.1.0**	⌘

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

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

Title:	⌘	Maximise the reuse of ITU-T CMIP event report management functions		
Source:	⌘	SA5		
Work item code:	⌘	OAM-NIM	***Date:*** ⌘	30/11/2001
Category:	⌘	**C**	***Release:*** ⌘	REL-5

Use one of the following categories:
- ***F*** *(correction)*
- ***A*** *(corresponds to a correction in an earlier release)*
- ***B*** *(addition of feature),*
- ***C*** *(functional modification of feature)*
- ***D*** *(editorial modification)*

Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

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

Reason for change:	⌘	Changes to the Notification IRP CMIP Solution Set in order to Maximise the reuse of ITU-T CMIP event report management functions
Summary of change:	⌘	Modification of the mapping tables and GDMO definitions
Consequences if not approved:	⌘	Fail to maximise reuse of ITU-T CMIP event report management functions

Clauses affected:	⌘	4, 5, 6		
Other specs affected:	⌘	☐ Other core specifications	⌘	
		☐ Test specifications		
		[X] O&M Specifications		32.302
Other comments:	⌘	If this "parent" CR is approved then also the attached "child" CR can be approved: Rel-5 CR 32.302 (S5-010773)		

4 Basic aspects

The present document provides~~defines all~~ the GDMO and ASN.1 definitions necessary to implement the Notification IRP ~~Information Service~~ for the CMIP interface. The definitions provided in the present document are employed by~~the base to implement~~ any other IRP that includes event reporting and/or management of event reporting.

~~The terms “manager/agent” are applied in the present document to mean “IRP Manager/IRP Agent” introduced in 3GPP TS 32.302 [3].~~

4.1 Architectural aspects

The architecture of the Notification IRP CMIP Solution Set is adapted as much as possible to the event reporting management model as defined in ITU-T Rec. X.734 [10].

~~This CMIP Notification IRP is based, as much as possible, on the ITU-T TMN architecture, as defined through the ITU-T X.700 Recommendations series.~~

4.1.1 Event report management function in ITU-T

4.1.1.1 Event report management model

According to the event reporting management model specified in ITU-T Rec. X.734 [10] each managed object may emit notifications (potential event reports). Conceptually, these potential event reports are distributed to all event forwarding discriminators (EFDs) that are instantiated in the IRPAgent. The event forwarding discriminators process the potential event reports to determine which event reports are to be forwarded to a particular destination. The conditions event reports must satisfy in order to be forwarded are specified by the discriminator construct. This is a set of one or more assertions about the presence or value of attributes of the potential event report.

Operational and administrative states are defined for event forwarding discriminators. The operational state has two possible values: enabled and disabled. In the enabled state the discriminator processes the potential event reports. In the disabled state potential event reports are not processed. The administrative states defined are locked and unlocked. When the state is changed from unlocked to locked forwarding of event reports is suspended. When the administrative state is changed from locked to unlocked event forwarding is resumed.

4.1.1.2 Event forwarding discriminator management

The event forwarding discriminator is a managed object. Event reporting is controlled by performing operations on these objects. The required management operations are defined in ITU-T Rec. X.710 [5].

In order to initiate the transmission of event reports an event forwarding discriminator has to be created in the IRPAgent. For this purpose the CMISE M-CREATE service is used. In order to terminate the transmission the discriminator has to be deleted (M-DELETE). The filtering mechanism may be changed by modifying the discriminator construct attribute. This operation is requested by M-SET. The transmission may be suspended and resumed by changing the administrative state from unlocked to locked and vice versa. Also for modifying the administrative state the M-SET service is used.

4.1.1.3 Definition of notifications

ITU-T Rec. X.734 [10] does not define any specific notifications. Instead, any object of the IRPAgent that shall have the capability to emit notifications must have the GDMO and the supporting ASN.1 syntax definition of these notifications included in the definition of its managed object class. More specifically, whereas the present document defines the managed objects and operations for the event reporting function the other IRPs must specify the information to be carried in the notifications.

The event reports are sent from the IRPAgent to the IRPManager using the CMISE service M-EVENT-REPORT, defined in ITU-T Rec. X.710 [5] and ITU-T Rec. X.711 [6].

4.1.2 Mediation between the concepts of the Notification IRP IS and ITU-T

The Notification IRP Information Service defines several operations allowing the IRPManager to control the event reporting: subscribe, unsubscribe, suspend subscription, resume subscription, change filter, get subscription status, get subscription identifiers.

The subscription-related operations of the Notification IRP (subscribe, unsubscribe, suspendSubscription, resumeSubscription, changeSubscriptionFilter, getSubscriptionStatus, getSubscriptionIds) are mapped into CMISE services. The remaining operations of the Notification IRP (getNotificationCategories, getNotificationIRPVersion, getOperationProfile, getNotificationProfile) allowing the IRPManager to retrieve informations pertaining to the Notification IRP are implemented as GDMO actions by a special managed object in the IRPAgent.

The EFDs are hence directly controlled by the IRPManager. On Itf-N are invoked CMISE services when EFDs are managed and GDMO actions when the IRPManager retrieves information about the Notification IRP.

~~4.1.1 Notifications~~

~~The Notifications messages are sent from the Agent to the Manager using the CMISE service M-EVENT-REPORT, defined in ITU-T Recommendation X.710 [5] and ITU-T Recommendation X.711 [6].~~

~~Any object of the Agent that sends a specific notification to the Manager needs to have, in its Managed Object Class (MOC) Definition, the GDMO definition of that specific "Notification" and the supporting ASN.1 syntax definition. The present document does not define any specific Notification. The specific Notifications are defined in other "CMIP IRP Solution Sets", as necessary (e.g. the alarm notifications are defined in CMIP Alarm IRP Solution Set).~~

~~4.1.2 Event reporting management~~

~~In the higher level (protocol independent) description of the Notification IRP Information Service, the event reporting is managed (by the Manager) by means of several operations: subscribe, unsubscribe, suspend, resume subscription, change filter, etc. Most of these operations require the "subscription identifier" parameter to easy the handling of multiple subscriptions.~~

~~In the ITU-T TMN architecture the event reporting is managed by means of the MOC Event Forwarding Discriminator (EFD), which is instantiated on the Agent and is controlled by the Manager, by means of CMISE services (M-CREATE, M-SET, etc.). There is no attribute in the EFD that corresponds to the "subscription identifier".~~

~~The mapping between the operations defined in the Notification IRP Information Service and the CMISE services applicable to the EFD is not one-to-one, therefore a mediation function is necessary. This mediation function can be located on the Manager or on the Agent. In the first case, the Manager should translate the subscription-related operations in a sequence of one or more CMISE services, it should assign a subscription identifier and it should handle the mapping between the subscription identifier and the EFDs.~~

~~In the second case this mediation is performed by the Agent and is based on the following points:~~

- ~~A new MOC (i.e. *notificationControl*) is defined to be instantiated on the IRP Agent. This MOC has the purpose to implement the operations defined in Notification IRP Information Service and to interact with the local EFD(s). The operations are implemented as Actions. There is a one-to-one mapping between the operations and the Actions.~~

- ~~The EFD defined in ITU-T Recommendation X.734 [10] and ITU-T Recommendation X.721 [7] is used for event reporting, however this EFD shall be controlled by the agent. In other words, it shall be created/deleted and its attributes shall be managed by the Agent, via *notificationControl* MOI.~~

- ~~The Manager shall interact with *notificationControl* MOI located on the IRP Agent to execute the subscription related Actions. It is responsibility of the *notificationControl* MOI to assign the "subscription identifier" and to handle the correspondence between the subscription identifiers, the EFDs and the *discriminatorConstruct* associated to each subscription.~~
  ~~It is not required that the Manager controls directly the EFD by means of CMISE services.~~

~~The second alternative is chosen. The rest of this Solution Set (SS) is based on this choice.~~

~~4.1.3 Subscription related operations~~

~~The operation that allows the Manager to receive notifications from the Agent is *subscribe*.~~

~~The IRP concept foresees in different operations a parameter *subscriptionId*, which is generated by the Agent as response to a *subscribe* request and unambiguously identifies a Manager subscription in the scope of the whole Agent. Therefore the Agent is required to maintain at any time a table of correspondence between every subscription and the related EFD instance.~~

~~When the forwarding of some notifications is not needed any more, the Manager may invoke an *unsubscribe* operation. In this case one or all subscriptions available for this Manager are cancelled, e.g. the Agent may implicitly delete also the related EFD instance(s).~~

~~The creation and deletion of EFD instances on the Manager-Agent interface is therefore "encapsulated", i.e. in the CMIP Solution Set the standardised M-CREATE and M-DELETE services (defined in ITU-T Recommendation X.710 [5] and ITU-T Recommendation X.711 [6]) are not directly used for the EFD management.~~

~~Note that only the mandatory EFD attributes (*destination* and *filter*, according to ITU-T Recommendation X.721 [7]) are supported by the *subscribe* operation.~~

~~To suspend/resume the forwarding of the notification towards a manager, the subscribed Manager shall use the *suspendSubscription/resumeSubscription* actions of *notificationControl*. These actions result in *locking/unlocking* the administrative state of the related EFD.~~

~~To change the filtering constraints associated to a subscription, the subscribed Manager shall use the *changeSbscriptionFilter* action of *notificationControl*. This action results in a change of the *discriminatorConstruct* of the related EFD.~~

4.2 Mapping

The semantics of the Notification IRP are defined in 3GPP TS 32.302 [3]. The definitions of the management information defined there are independent of any implementation technology and protocol. This clause maps these protocol independent definitions onto the equivalencies of the CMIP solution set of Notification IRP.

4.2.1 Mapping of Information Object Classes (IOC)

Table 1 maps the IOCs defined in the Notification IRP Information Service onto the corresponding Managed Object Classes ~~/ Attributes~~ defined in this CMIP Solution Set. The Managed Object Classes (MOC) are qualified as Mandatory (M) or Optional (O).

Table 1: Mapping of IOC

IOC of the Notification IRP Information Service	MOC or Attributes of the CMIP solution set	Qualifier
NotificationIRP	notificationControl	M
NtfSubscriber	--	
NtfSubscription	--	

4.2.2 Mapping of ~~of Interface and O~~operations

~~Table 2 maps the Interface/Operations defined in the Notification IRP Information Service onto the equivalent Actions of the notificationControl MOC of this CMIP Solution Set. The CMIP Actions are qualified as Mandatory (M) or Optional (O).~~

~~The CMIP Actions are based on he M-ACTION service of CMISE, defined in ITU-T Recommendation X.710 [5] and ITU-T Recommendation X.711 [6]).~~

Table 2 and Table 3 map the operations defined in the 3GPP TS 32.302 [3] (Notification IRP: Information Service) and 3GPP TS 32.312 [12] (Generic IRP Management: Information Service) onto corresponding CMISE services and GDMO actions. The operations are qualified as mandatory (M) or optional (O).

The CMISE services are defined in ITU-T Rec. X.710 [5].

Table 2: Mapping of ~~O~~operations of the Notification IRP IS

Interface	Operation	GDMO Action or CMISE of CMIP SS	Qualifier
NotificationIRPManagement	subscribe	M-CREATE (CMISE) Creation of an EFD	M
	unsubscribe	M-DELETE (CMISE) Deletion of an EFD	M
SubscriptionSuspendOperations	suspendSubscription	M-SET (CMISE) Modification of the administrative state of the EFD to locked	O
	resumeSubscription	M-SET (CMISE) Modification of the administrative state of the EFD to unlocked	O
SubscriptionFilterOperations	changeSubscriptionFilter	M-SET (CMISE) Modification of the discriminator construct in the EFD	O
SubscriptionStatusOperations	getSubscriptionStatus	M-GET (CMISE) Retrieval of EFD attributes	O
SubscriberManagement	getSubscriptionIds	M-GET (CMISE) Retrieval of the object instances of the EFDs having the specified destination attribute	O
IRPManagementOperations	getNotificationCategories	GetNotificationCategories	O

Table 3: Mapping of operations of the Generic IRP Management IS

Interface	Operation	GDMO Action of CMIP SS	Qualifier
GenericIRPVersionsOperations	getIRPVersion	getNotificationIRPVersion	M
GenericIRPProfileOperations	getOperationProfile	getOperationProfile	O
	getNotificationProfile	getNotificationProfile	O

~~Interface/Operations of the Notification IRP Information Service~~	~~GDMO Actions of notificationControl of CMIP solution set~~	~~Qualifier~~
~~NotificationIRPManagement/subscribe~~	~~subscribe~~	~~M~~
~~NotificationIRPManagement/unsubscribe~~	~~unsubscribe~~	~~M~~
~~SubscriberManagement/getSubscriptionIds~~	~~getSubscriptionIds~~	~~O~~
~~SubscriptionStatusOperations/getSubscriptionStatus~~	~~getSubscriptionStatus~~	~~O~~
~~SubscriptionFilterOperations/changeSubscriptionFilter~~	~~changeSubscriptionFilter~~	~~O~~
~~SubscriptionSuspendOperations/suspendSubscription~~	~~suspendSubscription~~	~~O~~ ~~Implemented if 'resume-Subscription' is implemented.~~
~~SubscriptionSuspendOperations/resumeSubscription~~	~~resumeSubscription~~	~~O~~ ~~Implemented if 'suspend-Subscription' is implemented.~~
~~IRPManagementOperations/getNotificationCategories~~	~~getNotificationCategories~~	~~O~~
~~GenericIRPVersionOperation/getIRPVersion~~	~~getNotificationIRPVersion~~	~~M~~
~~GenericIRPProfileOperation/getOperationProfile~~	~~getOperationProfile~~	~~O~~
~~GenericIRPProfileOperation/getNotificationProfile~~	~~getNotificationProfile~~	~~O~~

~~NOTE: The GenericIRPVersionOperation and GenericIRPProfileOperation are defined in [12].~~

4.2.3 Mapping of operation parameters

The tables in the following subclauses show the parameters of each operations defined in the Information Service described in TS 32.302 and their equivalence in this CMIP solution set.

The input parameters of the operations defined in TS 32.302 are mapped into "Action information" (see GDMO and ASN.1 definitions for more details).

The output parameters of the operations defined in TS 32.302 are mapped into "Action response" (see GDMO and ASN.1 definitions for more details).

4.2.3.1 Parameter ~~M~~mapping of the operation~~Parameters of~~ 'subscribe'

A manager subscribes to certain notifications by creating an appropriate EFD in the IRPAgent using the CMISE M-CREATE service.

The attribute list parameter of M-CREATE shall contain the values of the EFD attributes for destination and discriminatorConstruct.

The managed object instance of the created EFD is returned to the IRPManager in the M-CREATE success confirmation. According to ITU-T Rec. X.710 [5] this parameter has to be returned, if it is not supplied in the M-CREATE request.

Table ~~3~~4: Parameter ~~M~~mapping of the operation~~Parameters of~~ 'subscribe'

IS Parameter Name	IN/OUT	CMIP SS Equivalent	Qualifier
managerReference	IN	M-CREATE request parameter 'Attribute list': attribute identifier and value for the EFD 'destination' attribute	M
timeTick	IN	--	--
notificationCategories	IN	M-CREATE request parameter 'Attribute list': attribute identifier and value for the EFD 'discriminatorConstruct' attribute	O
filter	IN	M-CREATE request parameter 'Attribute list': attribute identifier and value for the EFD 'discriminatorConstruct' attribute	O
subscriptionId	OUT	M-CREATE success confirmation parameter 'Managed object instance'	M
status	OUT	status = OperationSucceeded The semantics of this status are conveyed by the emission of	M

		a M-CREATE success confirmation. status = OperationFailed The semantics of this status are conveyed by the emission of a M-CREATE failure confirmation.	

~~Operation parameters of the Information Services.~~	~~IN/OUT~~	~~CMIP Solution Set equivalences~~	~~Qualifier~~
~~managerReference~~	~~IN~~	~~managerReference~~	~~M~~
~~timeTick~~	~~IN~~	~~timeTick~~	~~O~~
~~notificationCategories~~	~~IN~~	~~notificationCatagoryList~~	~~O~~
~~filter~~	~~IN~~	~~filter~~	~~O~~
~~subscriptionId~~	~~OUT~~	~~subscriptionId~~	~~M~~
~~status~~	~~OUT~~	~~status~~	~~M~~
~~no equivalence~~		~~destination~~ ~~This information indicates a manager application which is designated to receive the concerned event reports issued by the related agent and is used to create the required EFD in the agent. It can be mapped onto the interface "notify" defined in the Information Service of the Notification IRP.~~	~~M~~

4.2.3.2 Parameter ~~M~~mapping of the operation~~Parameters of~~ 'unsubscribe'

The IRPManager can unsubscribe from receiving certain notifications by deleting the associated EFD using the M-DELETE service. The EFD to be deleted is identified by the M-DELETE parameters for the base object class and the base object instance.

The Notification IRP Information Service [3] specifies that a NtfSubscriber (IRPManager) may only delete subscriptions that are involved in a subscription relationship with the NtfSubsciber identified by the ManagerReference input parameter. This behaviour is mapped to a filtering mechanism in CMIP. The filter must specify an assertion on the EFD attribute 'destination' so that only EFDs whose destination attribute value specifies the IRPManager invoking this operation are selected for deletion.

In [3] it is also specified that all subscriptions made by the IRPManager specified in the managerReference input parameter shall be deleted when no subscriptionId is provided. This feature is mapped to a scoping and filtering mechanism. Scoped are all EFDs, selected by the filter are only those whose destination attribute specifies the invoking IRPManager.

Table ~~3~~4: Parameter ~~M~~mapping of the operation~~Parameters of~~ 'unsubscribe'

IS Parameter Name	IN/OUT	CMIP SS Equivalent	Qualifier
managerReference	IN	M-DELETE request parameters 'Scope' and 'Filter' Note: The filter parameter must specify an assertion selecting only EFDs whose destination attribute value specifies the IRPManager identified by managerReference.	M
subscriptionId	IN	M-DELETE request parameters 'Base object class' and 'Base object instance'	M

status	OUT	status = OperationSucceeded
		The semantics of this status are conveyed by the emission of a M-DELETE success confirmation.
		
		status = OperationFailed
		The semantics of this status are conveyed by the emission of a M-DELETE failure confirmation.
			M

~~Operation parameters of the Information Services.~~	~~IN/OUT~~	~~CMIP Solution Set equivalencies~~	~~Qualifier~~
~~managerReference~~	~~IN~~	~~managerReference~~	~~M~~
~~subscriptionId~~	~~IN~~	~~subscriptionId~~	~~M~~
~~status~~	~~OUT~~	~~status~~	~~M~~

4.2.3.3 Paramter ~~M~~mapping of the the operation~~Parameters of~~ ‘getSubscriptionIds’

The IRPManager may retrieve a list of its subscriptions using the M-GET service. For this purpose the M-GET parameter ‘Filter’ must specify an assertion selecting only EFDs whose destination attribute value specifies the IRPManager identified by managerReference. The object identifiers of the selected EFDs are returned in the M-GET response parameter ‘Managed object instance’. The attributes selected in the M-GET request parameter ‘Attribute identifier list’ and the values returned in the parameter ‘Attribute list’ are of no interest.

Table ~~4~~5: Parameter ~~M~~mapping of the operation~~Parameters of~~ ‘getSubscriptionIds’

IS Parameter Name	IN/OUT	CMIP SS Equivalent	Qualifier
managerReference	IN	M-GET request parameters ‘Base object class’, ‘Base object instance’, ‘Scope’ and ‘Filter’ Note: The filter parameter must specify an assertion selecting only EFDs whose destination attribute value specifies the IRPManager identified by managerReference.	M
subscriptionIdSet	OUT	M-GET response parameter ‘Managed object instance’	M

status	OUT	status = OperationSucceeded
		The semantics of this status are conveyed by the emission of a M-GET success confirmation.
		
		status = OperationFailed
		The semantics of this status are conveyed by the emission of a M-GET failure confirmation.
			M

~~Operation parameters of the Information Services.~~	~~IN/OUT~~	~~CMIP Solution Set equivalences~~	~~Qualifier~~
~~managerReference~~	~~IN~~	~~managerReference~~	~~M~~
~~subscriptionIdSet~~	~~OUT~~	~~subscriptionIdList~~	~~M~~
~~status~~	~~OUT~~	~~status~~	~~M~~

4.2.3.4 Parameter ~~M~~mapping of the operation~~Parameters of~~ ‘getSubscriptionStatus’

The status of an EFD may be retrieved by the IRPManager by reading the attribute values of the EFD. For this purpose the CMIS service M-GET is used.

The emission of certain notifications is suspended when the administrative state of the corresponding EFD is locked. In the unlocked state notifications are forwarded to the IRPManager.

Table ~~5~~6: Parameter ~~M~~mapping of the operation~~Parameters of~~ ‘getSubscriptionStatus’

IS Parameter Name	IN/OUT	CMIP SS Equivalent	Qualifier
subscriptionId	IN	M-GET request parameters ‘Base object class’ and ‘Base object instance’	M
notificationCategoryList	OUT	--	--
filterInEffect	OUT	M-GET response parameter ‘Attribute list’: attribute identifier and value for the EFD ‘discriminatorConstruct’ attribute	M
subscriptionStatus	OUT	M-GET response parameter ‘Attribute list’: attribute identifier and value for the EFD ‘administrativeState’ attribute	O

		administrativeState
		locked = suspended
		unlocked = not suspended/resumed	

timeTick	OUT	--	--

status	OUT	status = OperationSucceeded
		The semantics of this status are conveyed by the emission of a M-GETsuccess confirmation.
		
		status = OperationFailed
		The semantics of this status are conveyed by the emission of a M-GET failure confirmation.
			M

~~Operation parameters of the Information Services.~~	~~IN/OUT~~	~~CMIP Solution Set equivalences~~	~~Qualifier~~
~~subscriptionId~~	~~IN~~	~~subscriptionId~~	~~M~~
~~notificationCategoryList~~	~~OUT~~	~~notificationCategoryList~~	~~M~~
~~filterInEffect~~	~~OUT~~	~~filterInEffect~~	~~M~~
~~subscriptionStatus~~	~~OUT~~	~~subscriptionStatus~~	~~O~~
~~timeTick~~	~~OUT~~	~~timeTick~~	~~O~~
~~status~~	~~OUT~~	~~status~~	~~M~~

4.2.3.5 Parameter ~~M~~mapping of the operation~~Parameters of~~ 'changeSubscriptionFilter'

The IRPManager may change the conditions to be satisfied by a potential event report before being forwarded by modifying the discriminator construct. The EFD is identified by the M-SET request parameters for the base object class and the base object instance. The new discriminator construct is specified in the M-SET request parameter 'Modification list'.

Table ~~6~~7: Parameter ~~M~~mapping of the operation~~Parameters of~~ 'changeSubscriptionFilter'

IS Parameter Name	IN/OUT	CMIP SS Equivalent	Qualifier
subscriptionId	IN	M-SET request parameters 'Base object class' and 'Base object instance'	M
filter	IN	M-SET request parameter 'Modification list': attribute identifier and value for the EFD 'discriminatorConstruct' attribute	M

status	OUT	status = OperationSucceeded
		The semantics of this status are conveyed by the emission of a M-SET success confirmation.
		
		status = OperationFailed
		The semantics of this status are conveyed by the emission of a M-SET failure confirmation.
			M

~~Operation parameters of the Information Services.~~	~~IN/OUT~~	~~CMIP Solution Set equivalences~~	~~Qualifier~~
~~subscriptionId~~	~~IN~~	~~subscriptionId~~	~~M~~
~~filter~~	~~IN~~	~~filter~~	~~M~~
~~status~~	~~OUT~~	~~status~~	~~M~~

4.2.3.6 Parameter ~~M~~mapping of the operation~~Parameters of~~ 'suspendSubscription'

The IRPManager may suspend the transmission of certain notifications by changing the administrative state of the corresponding EFD to locked. The M-SET service is used to request the change of the administrative state. The EFD is identified by the M-SET parameters for the base object class and the base object instance. The attribute to be modified and the new attribute value is specified in the M-SET request parameter 'Modification list'.

Table ~~7~~8: Parameter ~~M~~mapping of the operation~~Parameters of~~ 'suspendSubscription'

IS Parameter Name	IN/OUT	CMIP SS Equivalent	Qualifier
subscriptionId	IN	M-SET request parameters 'Base object class' and 'Base object instance'	M

status	OUT	status = OperationSucceeded
		The semantics of this status are conveyed by the emission of a M-SET success confirmation.
		
		status = OperationFailed
		The semantics of this status are conveyed by the emission of a M-SET failure confirmation.
			M

~~Operation parameters of the Information Services.~~	~~IN/OUT~~	~~CMIP Solution Set equivalences~~	~~Qualifier~~
~~subscriptionId~~	~~IN~~	~~subscriptionId~~	~~M~~
~~status~~	~~OUT~~	~~status~~	~~M~~

4.2.3.7 Parameter ~~M~~mapping of the operation~~Parameters of~~ 'resumeSubscription'

The IRPManager may resume the emission of certain notifications by changing the administrative state of the corresponding EFD to unlocked. The M-SET service is used to request the change of the administrative state. The EFD is identified by the M-SET request parameters for the base object class and the base object instance. The attribute to be modified and the new attribute value is specified in the M-SET request parameter 'Modification list'.

Table ~~8~~9: Parameter ~~M~~mapping of the operation~~Parameters of~~ 'resumeSubscription'

IS Parameter Name	IN/OUT	CMIP SS Equivalent	Qualifier
subscriptionId	IN	M-SET request parameters 'Base object class' and 'Base object instance'	M

status	OUT	status = OperationSucceeded
		The semantics of this status are conveyed by the emission of a M-SET success confirmation.
		
		status = OperationFailed
		The semantics of this status are conveyed by the emission of a M-SET failure confirmation.
			M

~~Operation parameters of the Information Services.~~	~~IN/OUT~~	~~CMIP Solution Set equivalences~~	~~Qualifier~~
~~subscriptionId~~	~~IN~~	~~subscriptionId~~	~~M~~
~~status~~	~~OUT~~	~~status~~	~~M~~

4.2.3.8 Paramter ~~M~~mapping of the operation~~Parameters of~~ 'getNotificationCategories'

Table ~~9~~10: Parameter ~~M~~mapping of the operation~~Parameters of~~ 'getNotificationCategories'

IS Parameter Name~~Operation parameters of the Information Services.~~	IN/OUT	CMIP SS Equivalent~~Solution Set equivalences~~	Qualifier
notificationCategoryList	OUT	notificationCategoryList	M
status	OUT	status	M

4.2.3.9 Parameter ~~M~~mapping of the operation~~Parameters of~~ ‘getIRPVersion’

Table 4~~11~~: Parameter ~~M~~mapping of the operationParameters of ‘getIRPVersion’

IS Parameter Name~~Operation parameters of the Information Services.~~	**IN/OUT**	**CMIP SS Equivalent~~Solution Set equivalences~~**	**Qualifier**
versionNumberSet	OUT	versionNumberList	M
status	OUT	status	M

4.2.3.10 Parameter Mapping of the Operation~~Parameters of~~ ‘getOperationProfile’

Table 4~~12~~: Parameter ~~M~~mapping of the operation~~of Parameters of~~ ‘getOperationProfile’

IS Parameter Name~~Operation parameters of the Information Services.~~	**IN/OUT**	**CMIP SS Equivalent~~Solution Set equivalences~~**	**Qualifier**
irpVersion	IN	irpVersionNumber	M
operationNameProfile	OUT	operationNameProfile	M
operationParameterProfile	OUT	operationParameterProfile	M
status	OUT	status	M

4.2.3.11 Parameter ~~M~~mapping of the opeartion~~Parameters of~~ ‘getNotificationProfile’

Table 4~~13~~: Parameter ~~M~~mapping of the operation~~Parameters of~~ ‘getNotificationProfile’

IS Parameter Name~~Operation parameters of the Information Services.~~	**IN/OUT**	**CMIP SS Equivalent~~Solution Set equivalences~~**	**Qualifier**
irpVersion	IN	irpVersionNumber	M
notificationNameProfile	OUT	notificationNameProfile	M
notificationParameterProfile	OUT	notificationParameterProfile	M
status	OUT	status	M

~~4.3Mapping of common notification parameters~~

4.2.4 Mapping of common notification parameters

The following table gives the mapping between the common information parameters of TS 32.302 onto the common parameters of M-EVENT-REPORT

Table ~~11~~15: Mapping of common notification parameters

Common Parameters	M-EVENT-REPORT Parameters	Qualifier
(see NOTE 1)	Invoke identifier	M
ManagedObjectClass	Managed object class	M
ManagedObjectInstance	Managed object instance	M
NotificationId	(see NOTE 2)	
EventTime	Event time	M
SystemDN	(see NOTE 3)	--
NotificationType	Event type	M

NOTE 1: There is no common parameter in IRP Notification that corresponds to Invoke Identifier defined in [5].

NOTE 2: The common parameter NotificationId is mapped onto notificationIdentifier ([7] [9]) which is no explicit M-EVENT-REPORT parameter. Instead, it is included in the M-EVENT-REPORT request parameter 'Event information'.~~The common parameter NotificationId is mapped onto notificationIdentifier ([7] [9]) which is not part of the M-EVENT-REPORT header, indeed it is one of the parameters of the event information.~~

NOTE 3: The common parameter SystemDN is conditional in TS 32.302 and is not used on the CMIP interfaces.

5 GDMO definitions

5.1 Managed Object Classes

5.1.1 notificationControl

```
notificationControl MANAGED OBJECT CLASS
    DERIVED FROM
        ”Rec. X.721 | ISO/IEC 10165-2 : 1992”:top;
    CHARACTERIZED BY
        ~~notificationControlBasicPackage,~~
        notificationIRPVersionPackage;
    CONDITIONAL PACKAGES
        notificationControlInfoPackage PRESENT IF "an instance supports it",
        notificationProfilePackage PRESENT IF "an instance supports it",
        ~~notificationSubscriptionFilterPackage PRESENT IF "an instance supports it",~~
        ~~notificationSubscriptionControlPackage PRESENT IF "an instance supports it ";~~

REGISTERED AS { ts32-304NotificationsObjectClass 1};
```

5.2 Packages

~~5.2.1 notificationControlBasicPackage~~

```
~~notificationControlBasicPackage PACKAGE~~
~~    BEHAVIOUR~~
~~        notificationControlBasicPackageBehaviour;~~
~~    ATTRIBUTES~~
~~        notificationControlId     GET;~~
~~    ACTIONS~~
~~        subscribe,~~
~~        unsubscribe;~~
~~REGISTERED AS { ts32-304NotificationsPackage 1};~~
```

~~notificationControlBasicPackageBehaviour BEHAVIOUR~~

~~DEFINED AS~~

~~“The object class *notificationControl* offers all functions defined in the Notification IRP IS enabling managers to subscribe to agents for getting notifications they are concerned. It enables the managers to control the behaviour and to retrieve the management information related to subscriptions~~

~~An instance of the 'notificationControl' MOC is identified by the value of the attribute 'notificationControlId'.~~

~~The action 'subscribe' is provides the Manager with the capability to establish the communication to an Agent in order to receive event reports.~~

~~The action 'unsubscribe' is invoked by the Manager to cancel one or all subscriptions to the Agent.”;~~

5.2.2 notificationControlInfoPackage

```
notificationControl~~Basic~~InfoPackage PACKAGE
    BEHAVIOUR
        notificationControl~~Basic~~InfoPackageBehaviour;
    ATTRIBUTES
        notificationControlId     GET;
        supportedNotificationCategories     GET;
    ACTIONS
```

```
        ~~subscribe,~~
        ~~unsubscribe;~~
        getNotificationCategories
REGISTERED AS { ts32-304NotificationsPackage 1};
```

notificationControlInfoPackageBehaviour BEHAVIOUR

DEFINED AS

"This package has been defined to allow the IRPManager to get information about its currently active subscriptions.

The attribute 'supportedNotificationCategories' indicates the categories of notifications supported by the current IRPAgent.

The action 'getNotificationCategories' provides the IRPManager with the capability to query the supported categories of notifications.

~~The action 'getSubscriptionStatus' is invoked by the Manager to get information about the status of the specified subscription.~~

~~The action 'getSubscriptionIds' allows the Manager to get all currently valid *subscriptionId* values assigned by the Agent to this Manager.~~";

5.2.3 notificationIRPVersionPackage

```
notificationIRPVersionPackage PACKAGE
    BEHAVIOUR
        notificationIRPVersionPackageBehaviour;
    ATTRIBUTES
        supportedNotificationIRPVersions      GET;
    ACTIONS
        getNotificationIRPVersion;
REGISTERED AS { ts32-304NotificationsPackage 3};
```

notificationIRPVersionPackageBehaviour BEHAVIOUR

DEFINED AS

"This package has been defined to allow the IRPManager to get information about the Notification IRP versions supported by the IRPAgent.

The attribute 'supportedNotificationIRPVersions' indicates all versions of the NotificationIRP currently supported by the IRPAgent.

The action 'getNotificationIRPVersion' is invoked by the IRPManager to get information about the NotificationIRP versions supported by the IRPAgent.";

5.2.4 notificationProfilePackage

```
notificationProfilePackage PACKAGE
    BEHAVIOUR
        notificationProfilePackageBehaviour;
    ACTIONS
        getOperationProfile,
        getNotificationProfile;

REGISTERED AS { ts32-304NotificationsPackage 4};
```

notificationProfilePackageBehaviour BEHAVIOUR

DEFINED AS

"This package has been defined to allow the IRPManager to get detailed information about the profile of Notification IRP.

The action 'getOperationProfile' is invoked by the IRPManager to get detailed information about the operations supported by Notification IRP.

The action 'getNotificationProfile' is invoked by the IRPManager to get detailed information about the notifications supported by Notification IRP.";

~~5.2.5 notificationSubscriptionFilterPackage~~

~~notificationSubscriptionFilterPackage PACKAGE~~
~~BEHAVIOUR~~
~~notificationSubscriptionFilterPackageBehaviour;~~
~~ACTIONS~~
~~changeSubscriptionFilter;~~

~~REGISTERED AS { ts32-304NotificationsPackage 5};~~

~~notificationSubscriptionFilterPackageBehaviour BEHAVIOUR~~

~~DEFINED AS~~

~~"This Package provides the Manager with the capability to change the subscription filter.~~

~~The action 'changeSubscriptionFilter' provides the Manager with the capability to change the active filter for the current subscription.";~~

~~5.2.6 notificationSubscriptionControlPackage~~

~~notificationSubscriptionControlPackage PACKAGE~~
~~BEHAVIOUR~~
~~notificationSubscriptionControlPackageBehaviour;~~
~~ACTIONS~~
~~suspendSubscription,~~
~~resumeSubscription;~~

~~REGISTERED AS { ts32-304NotificationsPackage 6};~~

~~notificationSubscriptionControlPackageBehaviour BEHAVIOUR~~

~~DEFINED AS~~

~~"This package provides the Manager with the capability to control the subscriptions.~~

~~The action 'suspendSubscription' is invoked by the Manager to suspend an active subscription.~~

~~The action 'resumeSubscription' is invoked by the Manager to resume a subscription previously suspended.";~~

5.3 Actions

~~5.3.1 changeSubscriptionFilter (O)~~

~~changeSubscriptionFilter **ACTION**~~
~~**BEHAVIOUR**~~
~~changeSubscriptionFilterBehaviour;~~
~~**MODE**~~
~~CONFIRMED;~~
~~**WITH INFORMATION SYNTAX**~~
~~TS32-304-4TypeModule.ChangeSubscriptionFilter;~~
~~**WITH REPLY SYNTAX**~~

~~TS32-304-4TypeModule.ChangeSubscriptionFilterReply;~~
~~**REGISTERED AS** { ts32-304NotificationsAction 1};~~

~~changeSubscriptionFilterBehaviour **BEHAVIOUR**~~

~~**DEFINED AS**~~

~~"A Manager invokes this action to change the active filter for the subscription specified with 'subscriptionId' in the request. The Agent will modify in the related EFD instance the value of the attribute *discriminatorConstruct* accordingly.~~

~~The 'Action information' contains the following data:~~

~~□ *subscriptionId*~~

~~This mandatory parameter identifies unambiguously the Manager subscription.~~

~~□ *filter*~~

~~This mandatory parameter is used to change the value of the attribute *discriminatorConstruct* of the EFD taking into account the additional information:~~

~~- Parameter *notificatioCategories* (as specified in the *subscribe* action)~~

~~- An insertion which discriminates all notifications containing at the beginning of the attribute *additionaText* the string'(ALIGNMENT'. (see TS 32.111-4 for more details).~~

~~The 'Action response' is composed of the following data:~~

~~□ *status*~~

~~It contains the results of this action. Possible values: noError (0), error (the value indicates the reason of the error).";~~

5.3.2 getNotificationCategories (O)

getNotificationCategories **ACTION**
BEHAVIOUR
getNotificationCategoriesBehaviour;
MODE
CONFIRMED;
WITH REPLY SYNTAX
TS32-304-4TypeModule.GetNotificationCategoriesReply;
REGISTERED AS { ts32-304NotificationsAction 2};

getNotificationCategoriesBehaviour **BEHAVIOUR**

DEFINED AS

~~" A manager~~ An IRPManager may invoke this action to query the categories of notifications supported by a concerned ~~agent~~IRPAgent. This action is irrelevant to any subscriptions. An ~~manager~~ IRPManagermay invoke this action before or after a subscribtion.

The 'Action response' is composed of the following data:

- *notificationCategoryList*

This parameter identifies a list of categories of notifications supported by the concerned ~~agent~~IRPAgent. A list containing no element, i.e. a NULL list means that the ~~agent~~ IRPAgent does not support any category of notification.

- *status*

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

5.3.3 getNotificationIRPVersion (M)

getNotificationIRPVersion **ACTION**
  BEHAVIOUR
    getNotificationIRPVersionBehaviour;
  MODE
    CONFIRMED;
  WITH REPLY SYNTAX
    TS32-304-4TypeModule.GetNotificationIRPVersionReply;
REGISTERED AS { ts32-304NotificationsAction 3};

getNotificationIRPVersionBehaviour **BEHAVIOUR**

DEFINED AS

"An IRPManager invokes this action to enquiry about the version of the Notification IRP the concerned IRPAgent supports.

The 'Action information' field contains no data:

The 'Action response' is composed of the following data:

- *versionNumbersList*

  It contains a list of versions supported by the concerned ~~agent~~ IRPAgent which are backwards compatible. A list containing no element, i.e. a NULL list means that the concerned ~~agent~~ IRPAgent doesn't support any version of the Notification IRP.

- *status*

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

5.3.4 getNotificationProfile (O)

getNotificationProfile **ACTION**
  BEHAVIOUR
    getNotificationProfileBehaviour;
  MODE
    CONFIRMED;
  WITH INFORMATION SYNTAX
    TS32-304-4TypeModule.IRPVersionNumber;
  WITH REPLY SYNTAX
    TS32-304-4TypeModule.GetNotificationProfileReply;
REGISTERED AS { ts32-304NotificationsAction 4};

getNotificationProfileBehaviour **BEHAVIOUR**

DEFINED AS

"An IRPManager invokes this action to enquiry about the notification profile (supported notifications and supported parameters) for this specific Notification IRP version.

The 'Action information' contains the following data:

- *irpVersionNumber*

  This mandatory parameter identifies a Notification 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 Notification 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).”;

5.3.5 getOperationProfile (O)

getOperationProfile **ACTION**
BEHAVIOUR
getOperationProfileBehaviour;
MODE
CONFIRMED;
WITH INFORMATION SYNTAX
TS32-304-4TypeModule.IRPVersionNumber;
WITH REPLY SYNTAX
TS32-304-4TypeModule.GetOperationProfileReply;
REGISTERED AS { ts32-304NotificationsAction 5};

getOperationProfileBehaviour **BEHAVIOUR**

DEFINED AS

“An IRPManager invokes this action to enquiry about the operation profile (supported operations and supported parameters) for this specific Notification IRP version.

The 'Action information' contains the following data:

- *irpVersionNumber*

  This mandatory parameter identifies a Notification 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).”;

~~5.3.6 getSubscriptionIds (O)~~

~~getSubscriptionIds **ACTION**~~
~~**BEHAVIOUR**~~
~~getSubscriptionIdsBehaviour;~~

~~**MODE**~~
~~CONFIRMED;~~
~~**WITH INFORMATION SYNTAX**~~
~~TS32-304TypeModule.GetSubscriptionIds;~~
~~**WITH REPLY SYNTAX**~~
~~TS32-304TypeModule.GetSubscriptionIdsReply;~~
~~**REGISTERED AS** { ts32-304NotificationsAction 6};~~

~~getSubscriptionIdsBehaviour **BEHAVIOUR**~~

~~**DEFINED AS**~~

~~"A Manager invokes this action to query all currently valid *subscriptionId* values assigned by Agent to this Manager as result of previous *subscribe* operations triggered by this Manager.~~

~~The 'Action information' field contains the following data:~~

- ~~*managerReference*~~

  ~~This parameter identifies unambiguously the Manager invoking the current operation.~~

~~The response of this action is composed of the following data:~~

- ~~*subscriptionIdList*~~

  ~~This parameter identifies all *subscriptionId* currently valid for the Manager invoking this operation. The value of this parameter is NULL, if the Manager did not yet subscribed to that Agent or the Manager lost all subscription-related information.~~

- ~~*status*~~

  ~~It contains the results of this action. Possible values: noError (0), error (the value indicates the reason of the error).";~~

~~5.3.7 getSubscriptionStatus (O)~~

~~getSubscriptionStatus **ACTION**~~
~~**BEHAVIOUR**~~
~~getSubscriptionStatusBehaviour;~~
~~**MODE**~~
~~CONFIRMED;~~
~~**WITH INFORMATION SYNTAX**~~
~~TS32-304TypeModule.GetSubscriptionStatus;~~
~~**WITH REPLY SYNTAX**~~
~~TS32-304TypeModule. GetSubscriptionStatusReply;~~
~~**REGISTERED AS** { ts32-304NotificationsAction 7};~~

~~getSubscriptionStatusBehaviour **BEHAVIOUR**~~

~~**DEFINED AS**~~

~~"A manager invokes this action to query the status of the current subscription, identified by means of the *subscriptionId* value, returned by the Agent in the *subscribe* operation.~~

~~Some subscription status values relate to attributes of the EFD instance created by the manager within the agent, while other parameters refer to properties of the Manager-Agent communication.~~

~~The 'Action information' field contains the following data:~~

- ~~*subscriptionId*~~

  ~~This mandatory parameter identifies unambiguously the Manager subscription.~~

~~The response of this action is composed of the following data:~~

- ~~*notificationCategoryList*~~

  ~~This parameter identifies the categories of notifications supported in the current subscription. If the parameter value is NULL, all notification categories supported by the Agent are emitted towards the Manager.~~

- ~~*filterInEffect*~~

  ~~This parameter specifies the current *discriminatorConstruct* value of the EFD instance used by the Agent in the communication with the Manager. The value NULL means that no filter constraint applies to the notifications generated by the Agent.~~

- ~~*subscriptionStatus*~~

  ~~This optional parameter specifies if the current subscription is in the state 'suspended' or not.~~

- ~~*timeTick*~~

  ~~This optional parameter identifies the value of a timer controlled by the Agent for the supervision of the current subscription. The value is set by the Manager in the *subscribe* operation.~~

- ~~*status*~~

  ~~It contains the results of this action. Possible values: noError (0), error (the value indicates the reason of the error).";~~

~~5.3.8 resumeSubscription (O)~~

```
~~resumeSubscription ACTION~~
~~    BEHAVIOUR~~
~~        resumeSubscriptionBehaviour;~~
~~    MODE~~
~~        CONFIRMED;~~
~~    WITH INFORMATION SYNTAX~~
~~        TS32-304TypeModule.ResumeSubscription;~~
~~    WITH REPLY SYNTAX~~
~~        TS32-304TypeModule.ResumeSubscriptionReply;~~
~~REGISTERED AS { ts32-304NotificationsAction 8};~~
```

~~resumeSubscriptionBehaviour **BEHAVIOUR**~~

~~**DEFINED AS**~~

~~"A Manager invokes this action to resume a subscription previously suspended. The Agent will set to 'unlocked' the value of the attribute *administrativeState* of the EFD instance related to the subscription specified in the Manager request. Therefore the forwarding of notifications according to the current filter (*discriminatorConstruct* attribute value) is possible again.~~

~~The 'Action information' field contains the following data:~~

- ~~*subscriptionId*~~

  ~~This mandatory parameter identifies unambiguously the Manager subscription which shall be resumed.~~

~~The 'Action response' is composed of the following data:~~

- ~~*status*~~

  ~~It contains the results of this action. Possible values: noError (0), error (the value indicates the reason of the error).";~~

~~5.3.9 subscribe (M)~~

~~subscribe **ACTION**~~

~~**BEHAVIOUR**~~
~~subscribeBehaviour;~~
~~**MODE**~~
~~CONFIRMED;~~
~~**WITH INFORMATION SYNTAX**~~
~~TS32-304TypeModule.Subscribe;~~
~~**WITH REPLY SYNTAX**~~
~~TS32-304TypeModule.SubscribeReply;~~
~~**REGISTERED AS** { ts32-304NotificationsAction 9};~~

~~subscribeBehaviour **BEHAVIOUR**~~

~~**DEFINED AS**~~

~~”A Manager invokes this action to establish a subscription to the Agent for the specified notifications.~~

~~In the context of the CMIP Solution Set for Notification IRP, the availability of at least one EFD instance is a necessary pre-requisite for the Manager to receive event reports from the Agent The *subscribe* action allows the Manager to specify parameters related to the Manager-Agent communication.~~

~~After receiving the *subscribe* request, the Agent defines an unambiguous *subscriptionId* value for the current subscription and, if necessary, creates a new EFD instance according to the parameters specified in the action request.~~

~~The 'Action information' contains the following data:~~

~~□ *managerReference*~~

~~This parameter identifies unambiguously the Manager invoking the current *subscribe* operation.~~

~~□ *destination*~~

~~This parameter identifies the destination to which event reports that have passed the filter conditions are sent. According to ITU-T X.721, 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*~~

~~This parameter defines the conditions a notification shall fulfil in order to be forwarded to the Manager.~~

~~□ *timeTick*~~

~~This optional parameter identifies the value of a timer controlled by the Agent for the supervision of the current subscription. The timer is reset every time the Manager invokes the *getSubscriptionStatus* action. If the timer expires, the Agent considers the communication with the current Manager as aborted and subsequently releases the resources allocated for this Manager (similar behaviour as in case of an *unsubscribe* action). In order to re-establish the communication, the Manager shall invoke again the *subscribe* action.~~

~~□ *notificationCategoryList*~~

~~This optional parameter identifies one or more types of notifications required in the current subscription. If the parameter value is NULL or absent, the Manager requires that all notification types supported by the Agent shall be emitted.~~

~~NOTE: The *discriminatorConstruct* of the EFD is composed taking into account the following information:~~

~~- Parameter *filter*~~

~~- Parameter *notificatioCategories*~~

~~- An insertion which discriminates all notifications containing at the beginning of the attribute *additionaText* the string '(ALIGNMENT'. (see TS 32.111-4 for more details).~~

~~The ‘Action response’ is composed of the following data:~~

~~*subscriptionId*~~

~~This parameter identifies unambiguously the current Manager subscription in the scope of the Agent and shall be used later only by the Manager invoking this action.~~

~~*status*~~

~~It contains the results of this action. Possible values: noError (0), error (the value indicates the reason of the error).”;~~

~~5.3.10 suspendSubscription (O)~~

```
~~suspendSubscription ACTION~~
~~    BEHAVIOUR~~
~~        suspendSubscriptionBehaviour;~~
~~    MODE~~
~~        CONFIRMED;~~
~~    WITH INFORMATION SYNTAX~~
~~        TS32-304TypeModule.SuspendSubscription;~~
~~    WITH REPLY SYNTAX~~
~~        TS32-304TypeModule.SuspendSubscriptionReply;~~
~~REGISTERED AS { ts32-304NotificationsAction 10};~~
```

~~suspendSubscriptionBehaviour **BEHAVIOUR**~~

~~**DEFINED AS**~~

~~”A Manager invokes this action to suspend an active subscription. The Agent will set to 'locked' the value of the attribute *administrativeState* of the EFD instance related to the subscription specified in the Manager request. The forwarding of notifications via the current EFD instance is not possible any more.~~

~~The 'Action information' field contains the following data:~~

~~*subscriptionId*~~

~~This mandatory parameter identifies unambiguously the Manager subscription which shall be suspended.~~

~~The ‘Action response’ is composed of the following data:~~

~~*status*~~

~~It contains the results of this action. Possible values: noError (0), error (the value indicates the reason of the error).”;~~

~~5.3.11 unsubscribe (M)~~

```
~~unsubscribe ACTION~~
~~    BEHAVIOUR~~
~~        unsubscribeBehaviour;~~
~~    MODE~~
~~        CONFIRMED;~~
~~    WITH INFORMATION SYNTAX~~
~~        TS32-304TypeModule.Unsubscribe;~~
~~    WITH REPLY SYNTAX~~
~~        TS32-304TypeModule.UnsubscribeReply;~~
~~REGISTERED AS { ts32-304NotificationsAction 11};~~
```

~~unsubscribeBehaviour **BEHAVIOUR**~~

~~**DEFINED AS**~~

~~”A Manager invokes this action to cancel a subscription to the Agent. For the CMIP solution set this may result in the deletion of the related EFD instance.~~

~~The 'Action information' contains the following data:~~

~~*managerReference*~~

~~This parameter identifies unambiguously the Manager invoking the current *unsubscribe* operation. In order to cancel a particular subscription, the Manager shall indicate additionally a specific *subscriptionId* value.~~

~~*subscriptionId*~~

~~This parameter identifies unambiguously a Manager subscription, established by means of a previous *subscribe* operation. If the parameter value is NULL, all current subscriptions of the Manager identified by means of the *managerReference* are cancelled, i.e. all related EFD instances may be deleted as well.~~

~~The ‘Action response’ is composed of the following data:~~

~~*status*~~

~~It contains the results of this action. Possible values: noError (0), error (the value indicates the reason of the error).”;~~

5.4 Attributes

5.4.1 notificationControlId

notificationControlId **ATTRIBUTE**
WITH ATTRIBUTE SYNTAX
TS32-304TypeModule.GeneralObjectId;
MATCHES FOR EQUALITY;
BEHAVIOUR notificationControlIdBehaviour;
REGISTERED AS { ts32-304NotificationsAttribute 1};

notificationControlIdBehaviour **BEHAVIOUR**

DEFINED AS

“This attribute names an instance of a ‘notificationControl’ object class.”;

5.4.2 supportedNotificationCategories

supportedNotificationCategories **ATTRIBUTE**
WITH ATTRIBUTE SYNTAX
TS32-304TypeModule. NotificationCategoryList;
MATCHES FOR
EQUALITY;
BEHAVIOUR
supportedNotificationCategoriesBehaviour;
REGISTERED AS { ts32-304NotificationsAttribute 2};

supportedNotificationCategoriesBehaviour **BEHAVIOUR**

DEFINED AS

”This attribute provides the information concerning the categories of notifications currently supported by the IRPAgent.”;

5.4.3 supportedNotificationIRPVersions

supportedNotificationIRPVersions **ATTRIBUTE**
WITH ATTRIBUTE SYNTAX

```
        TS32-304TypeModule.SupportedNotificationIRPVersions;
    MATCHES FOR
        EQUALITY;
    BEHAVIOUR
        supportedNotificationIRPVersionsBehaviour;
REGISTERED AS { ts32-304NotificationsAttribute 3};
```

supportedNotificationIRPVersionsBehaviour **BEHAVIOUR**

DEFINED AS

"This attribute provides the information concerning the NotificationIRP versions currently supported by the IRPAgent.";

6 ASN.1 definitions

```
TS32-304TypeModule {itu-t(0) identified-organization(4) etsi(0) mobileDomain(0) umts-Operation-Maintenance(3)
                   ts-32-304(304) informationModel(0) asn1Module(2) version1(1)}


   DEFINITIONS IMPLICIT TAGS ::=
   BEGIN

   --EXPORTS everything
   IMPORTS
   Destination, DiscriminatorConstruct
   FROM Attribute-ASN1Module {joint-iso-ccitt ms(9) smi(3) part2(2) asn1Module(2) 1}
   CMISFilter
   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) }

ts32-304Prefix          OBJECT IDENTIFIER ::= { baseNodeUMTS ts-32-304(304)}
ts32-304InfoModel       OBJECT IDENTIFIER ::= { ts32-304Prefix informationModel(0)}

ts32-304NotificationsObjectClass  OBJECT IDENTIFIER ::= { ts32-304InfoModel managedObjectClass(3)}
ts32-304NotificationsPackage      OBJECT IDENTIFIER ::= { ts32-304InfoModel package(4)}
ts32-304NotificationsAttribute    OBJECT IDENTIFIER ::= { ts32-304InfoModel attribute(7)}
ts32-304NotificationsAction       OBJECT IDENTIFIER ::= { ts32-304InfoModel action(9)}



   -- Start of 3GPP SA5 own definitions

   ErrorCauses ::= ENUMERATED
      {
      noError (0),                     -- operation / notification successfully performed
      ~~wrongSubscriptionId (1),         -- the value of the parameter subscriptionId is not known for the Agent~~
      ~~wrongManagedReference (2),       -- for the current Manager there is no subscription available~~
      notificationIRPVersionNotSupported (3),    -- Notification IRP version requested by NM not supported by
                                       IRPAgent
      wrongFilter (4),                 -- the value of the filter parameter is not valid
      wrongDestination (5),            -- the value of the destination parameter ~~(subscribe)~~ is not valid
      ~~duplicatedSubscription (6),      -- the current Manager already performed a subscription with the same~~
                                       ~~parameters~~
      ~~wrongTimeTick (7),               -- the value of the timeTick parameter (subscribe) is not valid~~
      ~~wrongNotificationCategory (8),   -- the notification category specified in the subscribe request is unknown~~
      unspecifiedErrorReason (255)     -- operation failed, specific error unknown
      }

   ~~ChangeSubscriptionFilter ::= SEQUENCE~~
   ~~   {~~
   ~~   subscriptionId          GraphicString,~~
   ~~   filter                  CMISFilter    -- ITU-T X.711~~
   ~~   }~~

   ~~ChangeSubscriptionFilterReply ::= SEQUENCE~~
   ~~   {~~
   ~~   status                  ErrorCauses~~
   ~~   }~~

   GeneralObjectId ::= INTEGER
```

```
GetNotificationCategoriesReply ::= SEQUENCE
    {
    notificationCategoryList    NotificationCategoryList,
    status                      ErrorCauses
    }

GetNotificationIRPVersionReply ::= SEQUENCE
    {
    versionNumbersList     SupportedNotificationIRPVersions,
    status                 ErrorCauses
    }

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

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

~~GetSubscriptionStatus ::= SEQUENCE~~
~~{~~
~~subscriptionId GraphicString~~
~~}~~

~~GetSubscriptionStatusReply ::= SEQUENCE~~
~~{~~
~~notificationCategoryList NotificationCategoryList,~~
~~filterInEffect CMISFilter, -- ITU-T X.711~~
~~subscriptionState SubscriptionState OPTIONAL,~~
~~timeTick INTEGER OPTIONAL,~~
~~status ErrorCauses~~
~~}~~

~~GetSubscriptionIds ::= SEQUENCE~~
~~{~~
~~managerReference INTEGER~~
~~}~~

~~GetSubscriptionIdsReply ::= SEQUENCE~~
~~{~~
~~subscriptionIdList SubscriptionIdList,~~
~~status ErrorCauses~~
~~}~~

```
IRPVersionNumber ::= GraphicString

NotificationCategory ::= ENUMERATED
    {
    alarm              (1),--the notification category defined in the alarm IRP
    basicCM            (2) --the notification category defined in the basic CM IRP
    bulkCM             (3) --the notification category defined in the bulk CM IRP
    }

NotificationCategoryList ::= SET OF NotificationCategory
```

NotificationList ::= SET OF NotificationName

NotificationName ::= GraphicString

OperationList ::= SET OF OperationName

OperationName ::= GraphicString

ParameterList ::= SET OF ParameterName

ParameterListOfList ::= SET OF ParameterList

ParameterName ::= GraphicString

~~**ResumeSubscription** ::= SEQUENCE~~
~~{~~
~~subscriptionId GraphicString~~
~~}~~

~~**ResumeSubscriptionReply** ::= SEQUENCE~~
~~{~~
~~status ErrorCauses~~
~~}~~

~~**Subscribe** ::= SEQUENCE~~
~~{~~
~~managerReference INTEGER,~~
~~destination Destination, -- ITU-T X.721~~
~~filter DiscriminatorConstruct, -- ITU-T X.721~~
~~timeTick INTEGER OPTIONAL,~~
~~notificationCategoryList NotificationCategoryList OPTIONAL~~
~~}~~

~~**SubscribeReply** ::= SEQUENCE~~
~~{~~
~~subscriptionId GraphicString,~~
~~status ErrorCauses~~
~~}~~

~~**SubscriptionIdList** ::= SET OF GraphicString~~

~~**SubscriptionState** ::= ENUMERATED~~
~~{~~
~~suspended (0),~~
~~notSuspended (1)~~
~~}~~

SupportedNotificationIRPVersions ::= SET OF IRPVersionNumber

~~**SuspendSubscription** ::= SEQUENCE~~
~~{~~
~~subscriptionId GraphicString~~
~~}~~

~~**SuspendSubscriptionReply** ::= SEQUENCE~~
~~{~~
~~status ErrorCauses~~
~~}~~

~~**Unsubscribe** ::= SEQUENCE~~
~~{~~

```
~~    managerReference    INTEGER,~~
~~    subscriptionId              GraphicString~~
~~    }~~

~~**UnsubscribeReply** ::= SEQUENCE~~
~~    {~~
~~    status            ErrorCauses~~
~~    }~~

END -- of TS32-304TypeModule
```