Source: **SA5 (Telecom Management)**

Title: **Rel-6 CR 32.102 Deletion of clauses in 32.102 that have been moved to 32.150/1/2**

Document for: **Decision**

Agenda Item: **7.5.3**

Doc-1st-Level	Spec	CR	R	Phase	Subject	Cat	Vers.	Doc-2nd-Level	Workitem
SP-040112	32.102	035	-	Rel-6	Deletion of clauses in 32.102 that have been moved to new Rel-6 TSs 32.150/1/2	F	6.1.0	S5-042151	OAM-AR

3GPP TSG-SA5 (Telecom Management) **S5-042151**
Meeting #37, Malaga, SPAIN, 23-27 Feb 2004

CR-Form-v7

CHANGE REQUEST

⌘ **32.102** **CR** **035** ⌘ **rev** **-** ⌘ Current version: **6.1.0** ⌘

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

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

Title:	⌘	Deletion of clauses in 32.102 that have been moved to new Rel-6 TSs 32.150/1/2
Source:	⌘	**SA5** (Michael.Truss@motorola.com)
Work item code:	⌘	OAM-AR
Date:	⌘	27/02/2004
Category:	⌘	**F**
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:	⌘	A number of clauses of 32.102 have been moved to a new specifications (32.150, 32.151 & 32.152) and as those specifications are submitted for apoproval the relevant clauses should now be removed from 32.102
Summary of change:	⌘	A reference is added in 32.102 to TS 32.150 and clauses 10.4-10.6 and Annexes C, D, E, F & G are removed
Consequences if not approved:	⌘	32.102 would contain a lot of material duplicated in 32.150, 32.151 and 32.152

Clauses affected:	⌘	2, 10.3, 10.4, 10.5, 10.6, Annexes C, D, E, F & G

Other specs affected: ⌘

Y	N	
	X	Other core specifications
	X	Test specifications
	X	O&M Specifications

⌘

Other comments: ⌘ This CR should only be approved if TS 32.150, 32.151 & 32.152 are also approved.

How to create CRs using this form:

Change in Clause 2

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] ITU-T Recommendation M.3010 (2000): "Principles for a Telecommunications management network".

[2] 3GPP TS 32.101: "Telecommunication management; Principles and high level requirements".

[3] Void.

[4] ITU-T Recommendation X.200 (1994): "Information technology – Open Systems Interconnection – Basic Reference Model: The basic model".

[5] 3GPP TS 32.150: “Integration Reference Point (IRP) Concept and Definitions”.

~~Void.~~

[6] Void.

[7] Void.

[8] Void.

[9] TMF GB910: "Smart TMN Telecom Operations Map (Release 2.1)". http://www.tmforum.org

[10] TMF GB909: "Smart TMN Technology Integration Map (Issue 1.1)". http://www.tmforum.org

[11] ITU-T Recommendation M.3013 (2000): "Considerations for a telecommunications management network".

[12] 3GPP TS 23.002: "Network architecture".

[13] 3GPP TS 23.101: "General UMTS Architecture".

[14] 3GPP TS 32.111 parts 1 to 4: "Telecommunication management; Fault Management;".

[15] OMG: "Unified Modelling Language Specification, Version 1.4, September 2001". http://www.omg.org/technology/documents/formal/uml.htm

End of Change in Clause 2

Change in Clause 10.3

10.3 Network infrastructure IRPs

When providing integrated management solutions for multi-vendor networks, there is a strong requirement that the NEs and the management solutions that go together with them are systems integratable.

It should be noted that these IRPs could be provided by either the NE, or the Element Manager (EM) or Sub-Network Manager (SNM) that goes together with the type of NE. There is actually not a clear distinction any more between NE and element management applications, mainly due to the increased processing capacity of the equipment platforms. Embedded Element Managers providing a web user interface is a common example of that.

These IRPs are introduced to ensure interoperability between Product-Specific Applications (PSA) and the Network & System Management Processes of the Network Manager (ref [2] & [9]) shown in the figure 10.3. These IRPs are considered to cover the most basic needs of task automation.

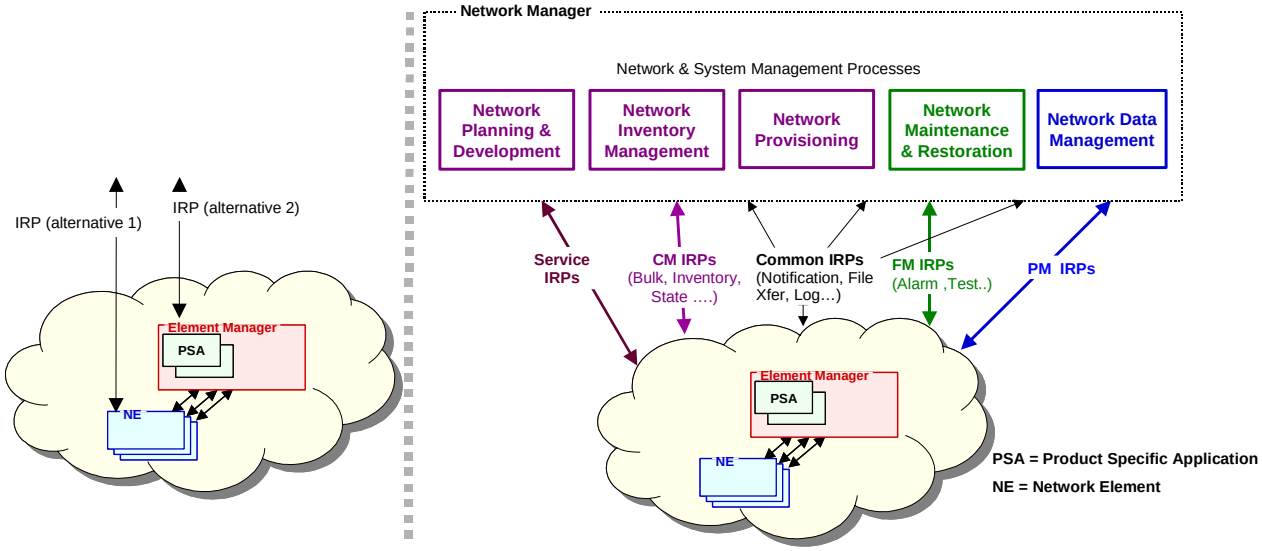


Figure 10.3: IRPs for application integration

The IRPs presented in figure 10.3 are just an example and do not reflect the exact set of IRPs defined by the 3GPP.

Taking one of the Common IRPs as an example, the Network & System Management Processes have similar need to receive notifications from various PSAs. The corresponding service is formalised as a *Notification IRP*.
It specifies: firstly, an interface through which subscriptions to different types of notifications can be set up (or cancelled), and secondly, common attributes for all notifications.

Further, applying a common *Name Convention for Managed Objects* is useful for co-operating applications that require identical interpretation of names assigned to network resources under management.

Further detail on the definition of IRPs can be found in TS 32.150 IRP Concepts and Definitions [5].

End of Change in Clause 10.3

Deletion of Clauses 10.4, 10.5 & 10.6

10.4 Defining the IRPs

It is important to accommodate more than one specific technology, as the technologies will change over time. Applications need to be future-proof. One fundamental principle for achieving this is to clearly separate the semantics of information definition from the protocols definitions (accessing the information) for the external interfaces.

The framework being used to define IRPs allows the implementation of user requirements for each management capability (e.g. configuration management), by modelling the information related to the resources to be managed and the way that the information may be accessed and manipulated. Such modelling is done in a way that is independent of the technology and distribution used in the implementation of a management system.

An IRP for a management capability is composed of three levels of specifications. The first level of IRP specification captures the **requirements**.

The second level of IRP specifications known as "**IRP Information Service**", specifies the **information** observable and controlled by management system's client, related to the network resources under management, in a technology independent way. It also specifies the semantics of the interactions used to carry applicable information.

The third level of IRP specification, known as **IRP Solution Set**, contains specification of the system in terms of interface technology choice (e.g. CMIP/GDMO, CORBA/IDL). In this type of specification, the syntax, rather than the semantics, is specified. At least one instance of a Solution Set is produced per interface technology supported.

The IRP methodology uses the following steps:

a) Capture the management requirements.

b) Specify the semantics of the information to describe the system. Trace back to item (a).

c) Specify the semantics of the interactions between the management system and its clients. Trace back to item (a).

d) Specify the syntaxes of the information and interactions identified in (b) and (c). The specification is technology dependent. Trace back to items (b) and (c).

As presented above, the Information Service document may contain two parts, the information related to the resources to be managed and the way that the information may be manipulated.

The first part defines the information types within a distributed system. It is in line with the Analysis phase of ITU-T M.3020. From the point of view of the Network Level modelling work it reflects the information aspects (including states and significant transitions) of the managed resources and the management services. It defines information object classes, the relationships between these object types, their attributes and states along with their permitted state transitions. It may also define the allowable state changes of one or more information objects. As recommended in M.3020, UML diagrams (class diagram, state diagram) are used to represent information when appropriate. This rest of the specification is described using an information description specified in natural language with appropriate label keywords (e.g. DEFINITION, ATTRIBUTE, CONSTRAINTS, etc…). A definition of the IS information template is provided in annex C.

Management service specific information objects may be created by subclassing from the objects in the basic network model, and extending them for that application. In this case, the new management service specific subclass may include other attributes, in addition to those defined in its superclass. Additional relationships and attributes may also be created as needed for that management service. Completely new objects can also be added.

The second part defines interfaces. Each interface contains one or more operations or one or more notifications that are made visible to management service users. An interface encapsulates information exchanged that is atomic in the sense that either all the information exchanged are visible (to management service users) or none. In addition, the specification of the information exchanged is in semantics only. No syntax or encoding can be implied. The operations or notifications are defined with their name, input and output parameters, pre and post conditions, raised exceptions and operation behaviour. These operation and notification specifications refer, through the utilisation of parameter matching, to the information objects. A definition of the IS operations/notifications template is provided in annex C.

The Solution Set contains the mapping of the information objects and interactions (if applicable) specified in the IS level specification, into their corresponding syntaxes of a particular chosen technology. The mapping is interface technology specific and satisfies scenarios where interfaces have been selected, according to mapping choices (driven

for example by system performance, development cost, time to market). The mapping is not always one-to-one.

General rules valid for all IRP Solution Sets are defined in Annex E. Rules for specific Solution Sets, such as CORBA, are defined in an Annex to the present document as well as within each of the Solution Set technologies used by 3GPP (as applicable).

Managed Object Classes as defined in a CMIP or CORBA Solution Set document represents a mapping into GDMO or IDL of Information Object Classes and other additional objects classes that can be introduced to support interfaces defined in the Information Service. Whether instances of Managed Object Classes are directly accessible or not may not be specified by IRP specifications.

Figure 10.4 shows an example of how an IRP can be structured (the Alarm IRP). Note that Figure 10.4 is only an example of what could be the Alarm IRP, the Alarm IRP specified in GPP TS 32.111 series [14] can be different.

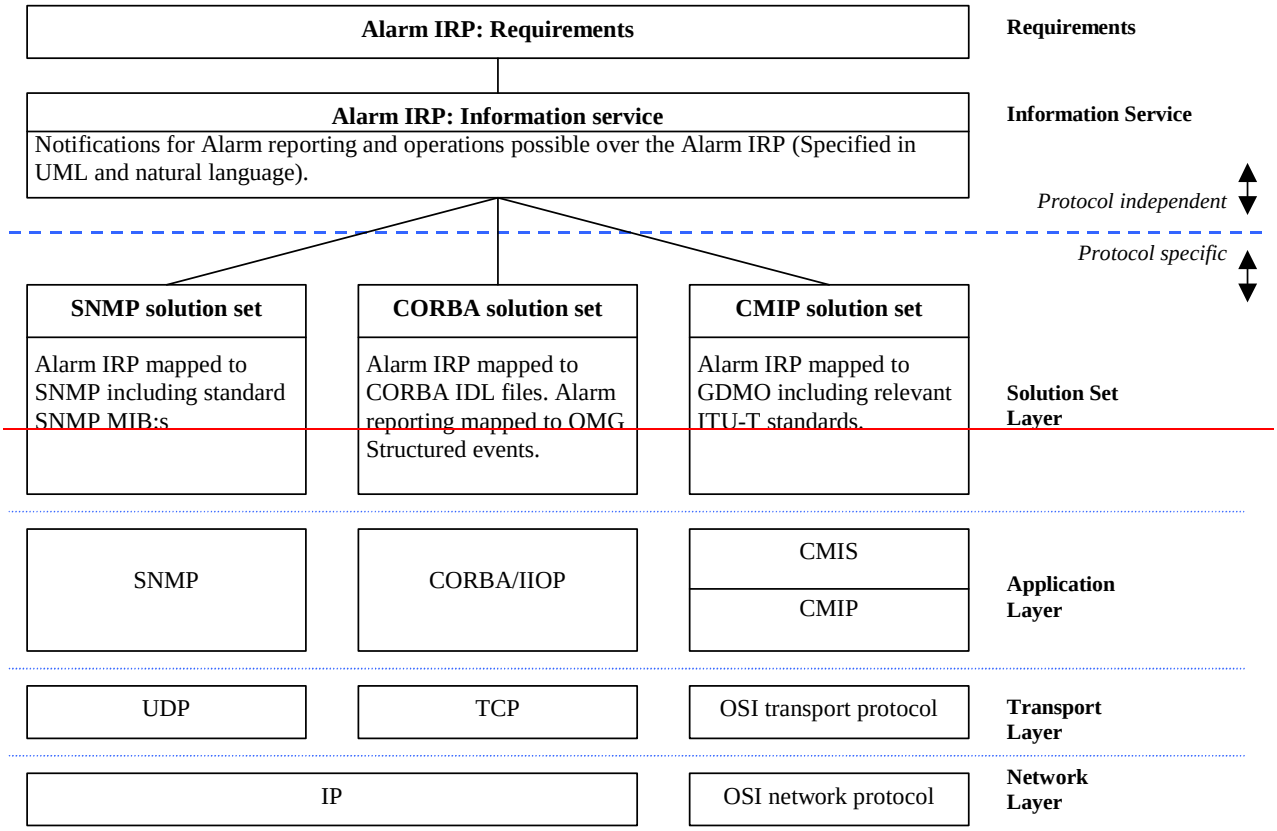


Figure 10.4: Example of an IRP (Alarm IRP)

10.5 Relationships between IS-level specifications

This subclause presents the target architecture of the SA5 IRP Information Models. This architecture is based on the concepts of level and partition of information. To achieve this, Information Object Classes (IOCs) and interfaces are defined and grouped into packages that can be related to each other through the *import* relationship.

Level means that the information models are structured in a way that enables re-utilization between levels, either through inheritance or through a traditional relationship between classes. Four levels are identified, namely:

A **Generic Network Resource Model**, also called "Generic NRM", which defines the information object classes that are independent of any 1/ protocol (e.g. CORBA / IDL, CMIP / GDMO, etc.) and 2/ "domain specific network" (e.g. UTRAN, GERAN, CN). This Network Resource Model contains definitions of the largest subset of information object classes that are common to all the Network Resources Models to be defined in SA5. This Network Resources Model is part of Level 1. For this Information Service, a number of solution sets may be provided;

A number of **Domain specific Network Resource Models**. Examples of these are: the CN Model, the UTRAN Model and the GERAN Model. They are part of Level 2. These Network Resource Models are specified in corresponding packages and import information object classes from the Generic Network Resources Model defined in Level 1. For each of these Information Services, a number of solution sets may be provided;

A number of **function specific ISs**. Such information services as the Basic CM IRP IS, the Notification IRP IS and the Alarm IRP IS are part of this level. They are part of Level 3. These Information Services are specified in corresponding packages and may import information object classes and interfaces defined in Level 1 and 2. For each of these Information Services, a number of solution sets may be provided;

A number of (interface technology independent) **Information Models**. Up to now, none of them have been defined. They will be part of Level 4. These Information Models are specified in corresponding packages and may import information object classes and interfaces defined both in Level 1, 2 and 3. An example of such Information Model could be a "UTRAN Alarm IM" (see Figure 10.5). For each of these Information Models, a number of solution sets may be provided;

These levels provide a means for separation of concerns and re-utilization.

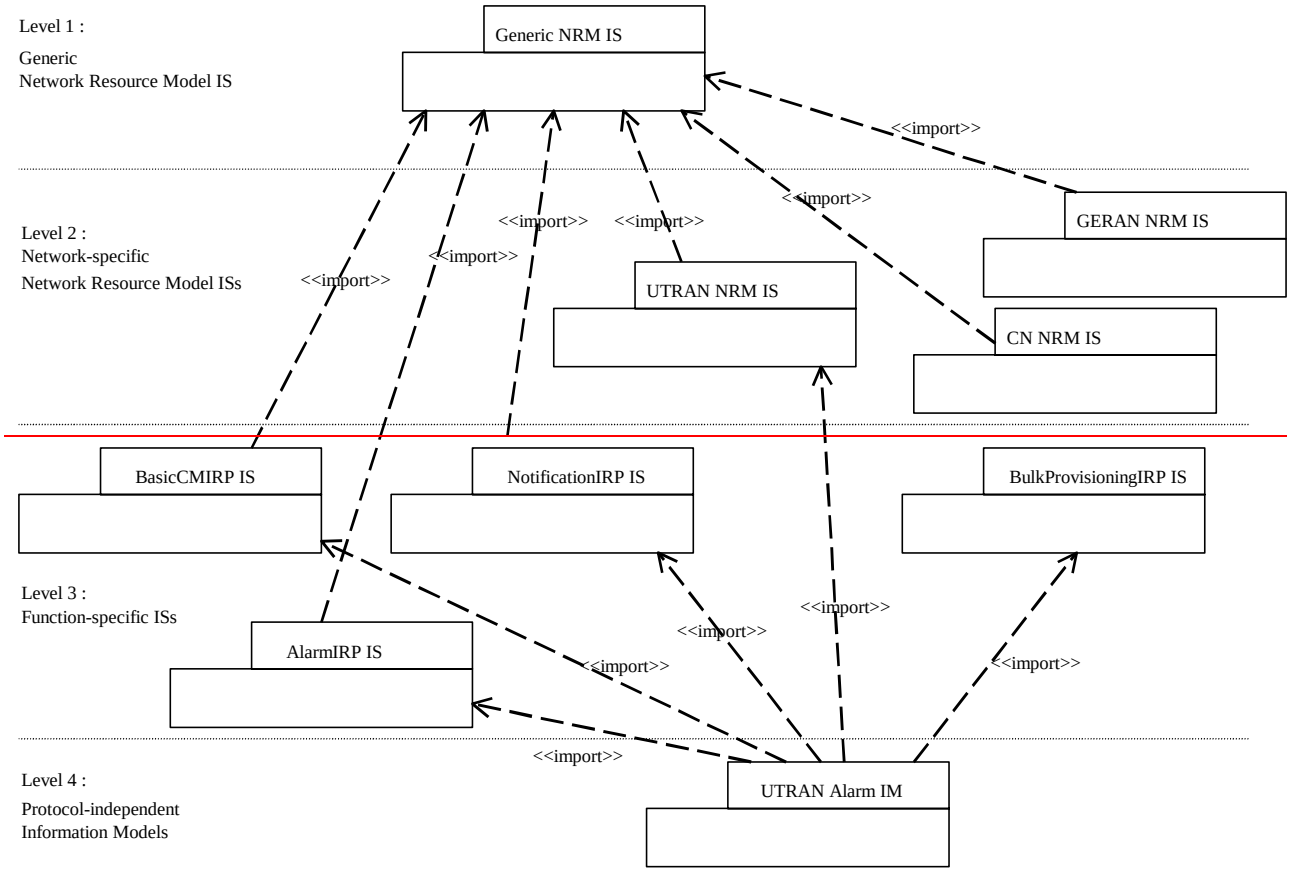
ISs shall be kept as simple as possible. To achieve this, information object classes and interfaces shall be grouped into packages. The grouping shall be based on semantics, i.e. information object classes and interfaces that participate in the definition of a given IRP should be gathered into a dedicated package. See further example(s) on this in Annex D.

Re-utilization of information specification contained in an IS previously specified shall be possible through the *import* relationship. The import relationship is a means for re-utilization : once a piece of information (i.e. an information object class, an attribute, a relationship or an interface) defined in an IS is imported in another IS, it is added to the name space of the importing IS. Then, the whole information available in a IS is made up of the information which is owned by the IS itself (i.e. defined in this document) plus the information which is imported from other IS(s). This imported information can then be utilised in the importing IS, for instance, through:

inheritance (e.g. any information object class defined at Levels 2 to 4 inherits from the information object class Top defined in the generic NRM at Level 1), either directly or indirectly;

relationship (e.g. any information object class defined at Levels 2 to 4 may have a containment relationship with the information object class IRPAgent defined in the generic NRM at Level 1).

An illustration of this architecture is provided in figure 10.5 below; it uses the UML diagrammatic conventions.



~~Figure 10.5: Specification architecture (not complete)~~

~~In order not to mix up the concept of "information object class" and "interface" with other concepts such as "managed object class" and "manager / agent interface", the former are labelled according to the UML notation capability (cf. stereotype). "Information object class" is defined as a stereotype of "Class" in the UML meta-model. As a consequence, information object classes defined in Information Models are labelled <<InformationObjectClass>>. Similarly, interfaces are labelled <<Interface>>. In annex C you can find an example of the inheritance between some ISs.~~

~~The following piece of information regarding the Semantics of the relationship "import" can be imported from other standard documents:~~

~~An information object class. The definition of the IOC, the attributes and the roles that the IOC plays in some relationships are imported. The import clause shall specify the TS number from which the IOC is imported and the name of the IOC;~~

~~An attribute. Two cases are valid:~~

- ~~An attribute definition. In this case, the attribute definition is imported. The import clause shall specify the TS number from which the attribute is imported and the name of the attribute;~~
- ~~An attribute reference within an IOC definition. In this case, the attribute definition is imported together with its qualifier within the specified IOC. The import clause shall specify the TS number from which the attribute is imported, the name of the IOC and the name of the attribute;~~

~~A relationship. The definition of the relationship is imported. The import clause shall specify the TS number from which the relationship is imported and the name of the relationship;~~

~~An interface. The definitions of the interface and all its operations or notifications are imported. The import clause shall specify the TS number from which the interface is imported and the name of the interface;~~

An operation or a notification. The definition of the operation / notification is imported. The import clause shall specify the TS number from which the operation / notification is imported, the name of the interface in which the operation / notification is defined and the name of the operation / notification;

A piece of information **must** always be imported from the TS where it is initially defined. It cannot be imported from any other.

10.6 Mandatory, Optional and Conditional qualifiers

This subclause defines a number of terms used to qualify the relationship between the "Information Service", the "Solution Sets" and their impact on the IRP implementations. The qualifiers defined in this clause are used to qualify IRPAgent behaviour only. This is considered sufficient for the specification of the IRPs.

Table 1 defines the meaning of the three terms Mandatory, Conditional and Optional when they are used to qualify the relations between operations, notifications and parameters specified in "Information Service" documents and their equivalents in Solution Set (SS) documents.

Table 1: Definitions of Mandatory, Optional and Conditional Used in Information Service Documents

	Mandatory (M)	**Conditional (C)**	**Optional (O)**
Operation and Notification	Each Operation and Notification shall be mapped to its equivalents in all SSs. Mapped equivalent shall be M.	Each Operation and Notification shall be mapped to its equivalents in at least one SS. Mapped equivalent can be M or O.	Each Operation and Notification shall be mapped to its equivalents in all SSs. Mapped equivalent shall be O.
Input and output parameter	Each parameter shall be mapped to one or more information elements of all SSs. Mapped information elements shall be M.	Each parameter shall be mapped to its equivalent in at least one SS. Mapped equivalent can be M or O.	Each parameter shall be mapped to its equivalent in all SSs. Mapped equivalent shall be O.
Information relationship	Each relationship shall be supported in all SS's.	Each relationship shall be supported in at least one SS.	Each relationship shall be supported in all SS's.
Information attribute	Each attribute shall be supported in all SS's.	Each attribute shall be supported in at least one SS.	Each attribute shall be supported in all SS's.

Table 2 defines the meaning of the two terms Mandatory and Optional when they are used to qualify the operations, parameters of operations, notifications and parameters of notifications in Solution Sets.

Table 2: Definitions of Mandatory and Optional Used in Solution Set Documents

Mapped SS Equivalent	**Mandatory**	**Optional**
Mapped notification equivalent	IRPAgent shall generate it.	IRPAgent may or may not generate it.
Mapped operation equivalent	IRPAgent shall support it.	IRPAgent may or may not support this operation. If the IRPAgent does not support this operation, the IRPAgent shall reject the operation invocation with a reason indicating that the IRPAgent does not support this operation. The rejection, together with a reason, shall be returned to the IRPManager.
input parameter of the mapped operation equivalent	IRPAgent shall accept and behave according to its value.	IRPAgent may or may not support this input parameter. If the IRPAgent does not support this input parameter and if it carries meaning (i.e. it does not carry no-information semantics), the IRPAgent shall reject the invocation with a reason (that it does not support the parameter). The rejection, together with the reason, shall be returned to the IRPManager.
Input parameter of mapped notify equivalent AND output parameter of mapped operation equivalent	IRPAgent shall generate it.	IRPAgent may generate it.

End of Deletion of Clauses 10.4, 10.5 & 10.6

Deletion of Annex C, D, E, F & G

~~Annex C (informative):~~
~~Information Service (IS) template~~

~~This annex contains the template to be used for the IRP Information Service TSs produced within the 3GPP TSG SA WG5. This template is based on the latest 3GPP template which **must** be used for any 3GPP Technical Specification.~~

~~The introductory clauses of the 3GPP template (from clause 1 to clause 3) are unchanged.~~

~~This template is numbered starting with "X", which in general, should correspond to clause 4 which is the beginning of the main text document. However, if there is a need for a specific IS to introduce additional clauses in the body, X may correspond to a number higher than 4. For an NRM IRP IS only clause X shall be used.~~

~~The conclusive clauses/annexes of the 3GPP template are unchanged.~~

~~X Information Object Classes (IOCs)~~

~~*"X" represents a clause number in the actual Information Service TS.*~~

~~X.1 Information entities imported and local labels~~

~~*This clause identifies a list of information entities (e.g. information object class, information relationship, information attribute) that have been defined in other specifications and that are imported in the present document. This includes information entities from other specifications imported for inheritance purpose. Each element of this list is a pair (label reference, local label).The label reference contains the name of the specification where it is defined, the type of the information entity and its name. The local label of imported information entities can then be used throughout the specification instead of the label reference.*~~

~~*This information is provided in a table. An example of such a table is given here below:*~~

~~Label reference~~	~~Local label~~
~~32.106-5 [10], information object class, Top~~	~~Top~~

~~X.2 Class diagram~~

~~X.2.1 Attributes and relationships~~

~~*This first diagram represents all information object classes defined in this IS with all their relationships and all their attributes. This diagram shall contain relationship names, role name and role cardinality. This shall be a UML compliant class diagram.*~~

~~*Characteristics (attributes, relationships) of imported information object classes need not to be repeated in the diagram. Information object classes should be defined using the stereotype <<InformationObjectClass>>.*~~

X.2.2 Inheritance

This second diagram represents the inheritance hierarchy of all information object classes defined in this IS. This diagram does not need to contain the complete inheritance hierarchy but shall at least contain the parent information object classes of all information object classes defined in the present document. By default, an information object class inherits from the information object class "top". This shall be a UML compliant class diagram.

Characteristics (attributes, relationships) of imported information object classes need not to be repeated in the diagram. Information object classes should be defined using the stereotype <<InformationObjectClass>>.

NOTE: some inheritance relationships presented in X.2.2 can be repeated in X.2.1 to enhance readability.

X.3 Information object classes definition

Each information object class is defined using the following structure.

X.3.a InformationObjectClassName

InformationObjectClassName is the name of the information object class.

"a" represents a number, starting at 1 and increasing by 1 with each new definition of an information object class.

X.3.a.1 Definition

The <definition> sub-clause is written in natural language. The <definition> sub-clause refers to the information object class itself. The characteristics related to the relationships that the object class can have with other object classes can't be found in the definition. The reader has to refer to relationships definition to find such kind of information. Information related to inheritance shall be precised here.

X.3.a.2 Attributes

The <attributes> sub-clause presents the list of attributes, which are the manageable properties of the object class. Each element is a tuple (attributeName, visibilityQualifier, supportQualifier, readQualifier, writeQualifier):

- *The visibilityQualifier indicates whether the attribute is public, private or IRPAgent Internal ("+"," ", and "%" respectively). The semantics of public and private are as per the UML specification. The semantic of IRPAgent Internal is defined within the 3GPP UML Repertoire.*
- *The supportQualifier indicates whether the attribute is Mandatory, Optional, Conditional or not supported ("M","O","C", or " ", respectively).*
- *The readQualifier indicates whether the attribute shall be readable by the IRPManager. The semantics for readQualifier is identical to supportQualifier, for "M, "O", and " ".*
- *The writeQualifier indicates whether the attribute shall be writeable by the IRPManager. The semantics for writeQualifier is identical to supportQualifier, for "M", "O", and " ".*

There is a dependency relationship between the supportQualifier and visibilityQualifier, readQualifier, and writeQualifier. The supportQualifier indicates the requirements for the support of the attribute. For any given attribute, regardless of the value of the supportQualifier, at least one of the reqdQualifier or writeQualifier must be "M". The implication of the "O" supportQualifier is that the attribute is optional, however the read and write qualifiers indicate how the optional attribute shall be supported, should the optional attribute be supported. Regardless of the supportQualifier, if an attribute is supported then it shall be supported in accordance with the specified visibilityQualifier.

Private or IRPAgent Internal attributes are per definition not readable by the IRPManager. Their readQualifier is hence always " ".

Private or IRPAgent Internal attributes are per definition not writable by the IRPManager. Their writeQualifier is hence always " ".

The readQualifier and writeQualifier of a supported attribute, that is public, may not be both " ".

The use of " " in supportQualifier is reserved for documenting support of attributes defined by an «Archetype» IOC. Attributes with a supportQualifier of " " are not implemented by the IOC that is realizing a subset of the attributes defined by the «Archetype». The readQualifier and writeQualifier are of no relevance in this case. However, a not supported attribute is neither readable nor writable. For this reason the readQualifier and writeQualifier shall be " " for unsupported attributes.

For any IOC that uses one or more attributes from an «Archetype», a separate table shall be used to indicate the supported attributes. This table is absent if no «Archetype» attributes are supported. For example, if a particular IOC has defined attributes (i.e. attributes not defined by an «Archetype») and encapsulates attributes from two «Archetype»s, then the totality of the attributes of said IOC will be contained in three separate tables.

This information is provided in a table. An example of such a table is given below:

Attribute name	Visibility	Support Qualifier	Read Qualifier	Write Qualifier
ntfSubscriptionId	+	M	M	O

Another example, where the support qualifier is "O" is given here below:

Attribute name	Visibility	Support Qualifier	Read Qualifier	Write Qualifier
ntfSubscriptionId	+	O	M	O

In this example, the ntfSubscriptionId is an optional attribute. If the implementation chose to support ntfSubscriptionId, then the said implementation is required to support read and may support write.

NOTE: This sub-clause does not need to be present when there is no attribute to define.

X.3.a.3 Attribute constraints

The <attribute constraints> sub-clause presents constraints between attributes that are always held to be true. Those properties are always held to be true during the lifetime of the attributes and in particular don't need to be repeated in pre or post conditions of operations or notifications.

NOTE: This sub-clause does not need to be present when there is no attribute constraints to define.

X.3.a.4 Relationships

The <relationship> sub-clause presents the list of relationships in which this class in involved. Each element is a relationshipName.

NOTE: This sub-clause is optional and may be avoided since all relationships are represented in the class diagram in clause.X.2.1.

X.3.a.5 State diagram

The <state diagram> sub-clause contains state diagrams. A state diagram of an information object class defines permitted states of this information object class and the transitions between those states. A state is expressed in terms of individual attribute values or a combination of attribute values or involvement in relationships of the information object class being defined. This shall be a UML compliant state diagram.

X.3.a.6 Notifications

The <notifications> sub-clause presents the list of notifications that can be emitted across the Itf-N, with "object class" and "object instance" parameters of the notification header of these notifications identifying an instance of the IOC defined by the encapsulating sub-clause (i.e. X.3.a). The presence of notifications in the present sub-clause (i.e. X.3.a.6) does not imply nor identify those notifications as being originated from an instance of the IOC defined by the encapsulating sub-clause (i.e. X.3.a).

This information is provided in a table. An example of such a table is given below:

Name	Qualifier	Notes
notifyAckStateChanged	See Alarm IRP (3GPP TS 32.111-2 [:])	
notifyAttributeValueChange	O	
notifyChangedAlarm	See Alarm IRP (3GPP TS 32.111-2 [:])	
notifyClearedAlarm	See Alarm IRP (3GPP TS 32.111-2 [:])	
notifyNewAlarm	See Alarm IRP (3GPP TS 32.111-2 [:])	
notifyObjectCreation	O	
notifyObjectDeletion	O	
...	...	

X.4 Information relationships definition

Each information relationship is defined using the following structure :

X.4.a InformationRelationshipName (supportQualifier)

InformationRelationshipName is the name of the information relationship followed by a qualifier indicating whether the relationship is Mandatory, Optional or Conditional (M, O, C)

"a" represents a number, starting at 1 and increasing by 1 with each new definition of an information relationship

X.4.a.1 Definition

The <definition> sub-clause is written in natural language.

X.4.a.2 Roles

The <roles> sub-clause identifies the roles played in the relationship by object classes. Each element is a pair (roleName, roleDefinition)

This information is provided in a table. An example of such a table is given here below :

Name	Definition
isSubscribedBy	This role represents the one who has subscribed

X.4.a.3 Constraints

The <constraints> sub-clause contains the list of properties specifying the semantic invariants that must be preserved on the relationship. Each element is a pair (propertyName, propertyDefinition). Those properties are always held to be true during the lifetime of the relationship and don't need to be repeated in pre or post conditions of operations or notifications.

This information is provided in a table. An example of such a table is given here below :

Name	Definition
inv_notificationCategories AllDistinct	"the notification categories contained in the ntfNotificationCategorySet attribute of ntfSubscription playing the role hasSubscription are all distinct from each other"

X.5 Information attributes definition

Each information attribute is defined using the following structure :

X.5.1 Definition and legal values

This sub-clause contains for each attribute being defined its name, its definition written in natural language and a list of legal values supported by the attribute.

In the case where the legal values can be enumerated, each element is a pair (legalValueName, legalValueDefinition), unless a legalValueDefinition applies to several values in which case the definition is provided only once. When the legal values cannot be enumerated, the list of legal values is defined by a single definition.

This information is provided in a table. An example of such a table is given here below :

Attribute Name	Definition	Legal Values
ntfSubscriptionId	It identifies uniquely a subscription	N/A
ntfSusbcriptionState	It indicates the activation state of a subscription	"suspended" : the subscription is suspended "notSuspended" : the subscription is active

X.5.2 Constraints

The <constraints> sub-clause indicates whether there are any constraints affecting attributes. Each constraint is defined by a pair (propertyName, propertyDefinition). PropertyDefinitions are expressed in natural language.

An example is given here below :

Name	Definition
inv_TimerConstraints	"ntfTimeTickTimer is lower than or equal to ntfTimeTick"

X.6 Particular information configurations

Some configurations of information are special or complex enough to justify the usage of a state diagram to clarify them. A state diagram in this clause defines permitted states of the system and the transitions between those states. A state is expressed in terms of a combination of attribute values constraints or involvement in relationships of one or more information object classes.

Y Interface Definition

"Y" represents a number, immediately following "X"

Y.1 Class diagram representing interfaces

Each interface is defined in the diagram. This shall be a UML compliant class diagram.

Interfaces are defined using a stereotype <<Interface>>. Each interface contains a set of either operations or notifications which are mandatory or either a single operation or a single notification which is optional. The support of an interface by an information object class is represented by a relationship between the 2 entities with a cardinality (1..1) if all the operations or notifications contained in the interface are mandatory, and (0..1) if the operation or notification contained in the interface is optional. On the class diagram, each operation and notification in an interface shall be qualified as "public" by the addition of a symbol "+" before each operation and notification.

Y.2 Generic rules

The following rules are relevant for all IS. They shall simply be copied as part of the template.

Rule 1: each operation with at least one input parameter supports a pre-condition valid_input_parameter which indicates that all input parameters shall be valid with regards to their information type. Additionally, each such

operation supports an exception operation_failed_invalid_input_parameter which is raised when pre-condition valid_input_parameter is false. The exception has the same entry and exit state.

Rule 2: Each operation with at least one optional input parameter supports a set of pre-conditions supported_optional_input_parameter_xxx where "xxx" is the name of the optional input parameter and the pre-condition indicates that the operation supports the named optional input parameter. Additionally, each such operation supports an exception operation_failed_unsupported_optional_input_parameter_xxx which is raised when (a) the pre-condition supported_optional_input_parameter_xxx is false and (b) the named optional input parameter is carrying information. The exception has the same entry and exit state.

Rule 3: each operation shall support a generic exception operation_failed_internal_problem which is raised when an internal problem occurs and that the operation cannot be completed. The exception has the same entry and exit state.

Y.b InterfaceName Interface

InterfaceName is the name of the interface

"b" represents a number, starting at 3 and increasing by 1 with each new definition of an interface

Each interface is defined by its name and by a sequence of operations or notifications as defined here below.

Each operation is defined using the following structure.

Y.b.a Operation OperationName (supportQualifier)

OperationName is the name of the operation followed by a qualifier indicating whether the operation is Mandatory, Optional or Conditional (M, O, C)

"a" represents a number, starting at 1 and increasing by 1 with each new definition of an operation

Y.b.a.1 Definition

The <definition> sub-clause is written in natural language.

Y.b.a.2 Input parameters

List of input parameters of the operation. Each element is a tuple (inputParameterName, supportQualifier, InformationType, inputParameterComment)

This information is provided in a table. An example of such a table is given here below :

Parameter Name	Qualifier	Information type	Comment
managerReference	M	ntfSubscriber.ntfManagerReference	It specifies the reference of IRPManager to which notifications shall be sent.

Y.b.a.3 Output parameters

List of output parameters of the operation. Each element is a tuple (outputParameterName, supportQualifier, MatchingInformation, outputParameterComment)

This information is provided in a table. An example of such a table is given here below :

Parameter Name	Qualifier	Matching Information	Comment
versionNumberSet	M	notificationIRP.irpversion	It indicates one or more SS version numbers supported by the notificationIRP.

Y.b.a.4 Pre-condition

A pre-condition is a collection of assertions joined by AND, OR, and NOT logical operators. The pre-condition must be held to be true before the operation is invoked. An example is given here below :

notificationCategoriesNotAllSubscribed OR notificationCategoriesParameterAbsentAndNotAllSubscribed

Each assertion is defined by a pair (propertyName, propertyDefinition). All assertions constituting the pre-condition are provided in a table. An example of such a table is given here below :

Assertion Name	**Definition**
notificationCategoriesNotAllSubscribed	"at least one notificationCategory identified in the notificationCategories input parameter is supported by IRPAgent and is not a member of the ntfNotificationCategorySet attribute of an ntfSubscription which is involved in a subscription relationship with the ntfSubscriber identified by the managerReference input parameter".
notificationCategoriesParameterAbsentAndNotAllSubscribed	"notificationCategories input parameter is absent and at least one notificationCategory supported by IRPAgent is not a member of the ntfNotificationCategorySet attribute of an ntfSsubscription which is involved in a subscription relationship with the ntfSubscriber identified by the managerReference input parameter"

Y.b.a.5 Post-condition

A post-condition is a collection of assertions joined by AND, OR, and NOT logical operators. The post-condition must be held to be true after the completion of the operation. When nothing is said in a post-condition regarding an information entity, the assumption is that this information entity has not changed compared to what is stated in the pre-condition. An example is given here below :

subscriptionDeleted OR allSubscriptionDeleted

Each assertion is defined by a pair (propertyName, propertyDefinition). All assertions constituting the post-condition are provided in a table. An example of such a table is given here below :

Assertion Name	**Definition**
subscriptionDeleted	"the ntfSubscription identified by subscriptionId input parameter is no more involved in a subscription relationship with the ntfSubscriber identified by the managerReference input parameter and has been deleted. If this ntfSubscriber has no more ntfSubscription, it is deleted as well."
allSubscriptionDeleted	"in the case subscriptionId input parameter was absent, the ntfSubscriber identified by the managerReference input parameter is no more involved in any subscription relationship and is deleted, the corresponding ntfSubscription have been deleted as well."

Y.b.a.6 Exceptions

List of exceptions that can be raised by the operation. Each element is a tuple (exceptionName, condition, ReturnedInformation, exitState))

Y.b.a.6.c exceptionName

ExceptionName is the name of an exception

"c" represents a number, starting at 1 and increasing by 1 with each new definition of an exception

This information is provided in a table. An example of such a table is given here below :

Exception Name	**Definition**
Ope_failed_existing_subscription	**Condition:** (notificationCategoriesNotAllSubscribed OR notificationCategoriesParameterAbsentAndNotAllSubscribed) not verified **Returned information:** output parameter status is set to OperationFailedExistingSubscription **Exit state:** Entry State

Each notification is defined using the following structure.

Y.b.a Notification NotificationName (supportQualifier)

NotificationName is the name of the notification followed by a qualifier indicating whether the notification is Mandatory, Optional or Conditional (M, O, C).

"a" represents a number, starting at 1 and increasing by 1 with each new definition of a notification

Y.b.a.1 Definition

The <definition> sub-clause is written in natural language.

Y.b.a.2 Input parameters

List of input parameters of the notification. Each element is a tuple (inputParameterName, supportQualifier and filteringQualifier, matchingInformation, inputParameterComment)

The filteringQualifier indicates whether the parameter of the notification can be filtered or not. Values are Yes (Y) or No (N). The matchingInformation refers to information in the state "toState".

This information is provided in a table. The column "Qualifiers" contains the two qualifiers supportQualifier and filteringQualifier separated by a comma. An example of such a table is given here below :

Parameter Name	Qualifiers	Matching Information	Comment
managerReference	M,Y	ntfSubscriber.ntfManagerReference	It specifies the reference of IRPManager to which notifications shall be sent.

Y.b.a.3 Triggering event

The triggering event for the notification to be sent is the transition from the information state defined by the "from state" sub-clause to the information state defined by the "to state" sub-clause.

Y.b.a.3.1 From state

This sub-clause is a collection of assertions joined by AND, OR, and NOT logical operators. An example is given here below :

alarmMatched AND alarmInformationNotCleared

Each assertion is defined by a pair (propertyName, propertyDefinition). All assertions constituting the state "from state" are provided in a table. An example of such a table is given here below :

Assertion Name	Definition
alarmMatched	The newly generated network alarm matches with one AlarmInformation (same values for eventType, probableCause, specificProblem attributes) in AlarmList.
alarmInformationNotCleared	The perceivedSeverity attribute of the matched AlarmInformation is not cleared

Y.b.a.3.2 To state

This sub-clause is a collection of assertions joined by AND, OR and NOT logical operators. When nothing is said in a to-state regarding an information entity, the assumption is that this information entity has not changed compared to what is stated in the from state. An example is given here below :

resetAcknowledgementInformation AND perceivedSeverityUpdated

Each assertion is defined by a pair (propertyName, propertyDefinition). All assertions constituting the state "to state" are provided in a table. An example of such a table is given here below :

Assertion Name	Definition
resetAcknowledgementInformation	The matched AlarmInformation identified in inv_alarmMatched in pre-condition has been updated according to the following rule : ackTime, ackUserId and ackSystemId are updated to contain no information; ackState is updated to "unacknowledged";
perceivedSeverityUpdated	The perceivedSeverity attribute of matched AlarmInformation identified in inv_alarmMatched in pre-condition has been updated.

Z Scenario

"Z" represents a number, immediately following "Y"

List of sequence diagrams each describing a possible scenario. This shall be a UML compliant sequence diagram. This is an optional clause.

Annex D (informative): Example of inheritance between ISs

Figure D.1 illustrates the architecture defined in clause 10.6 with a simplified example. Note that this figure is for illustration only.

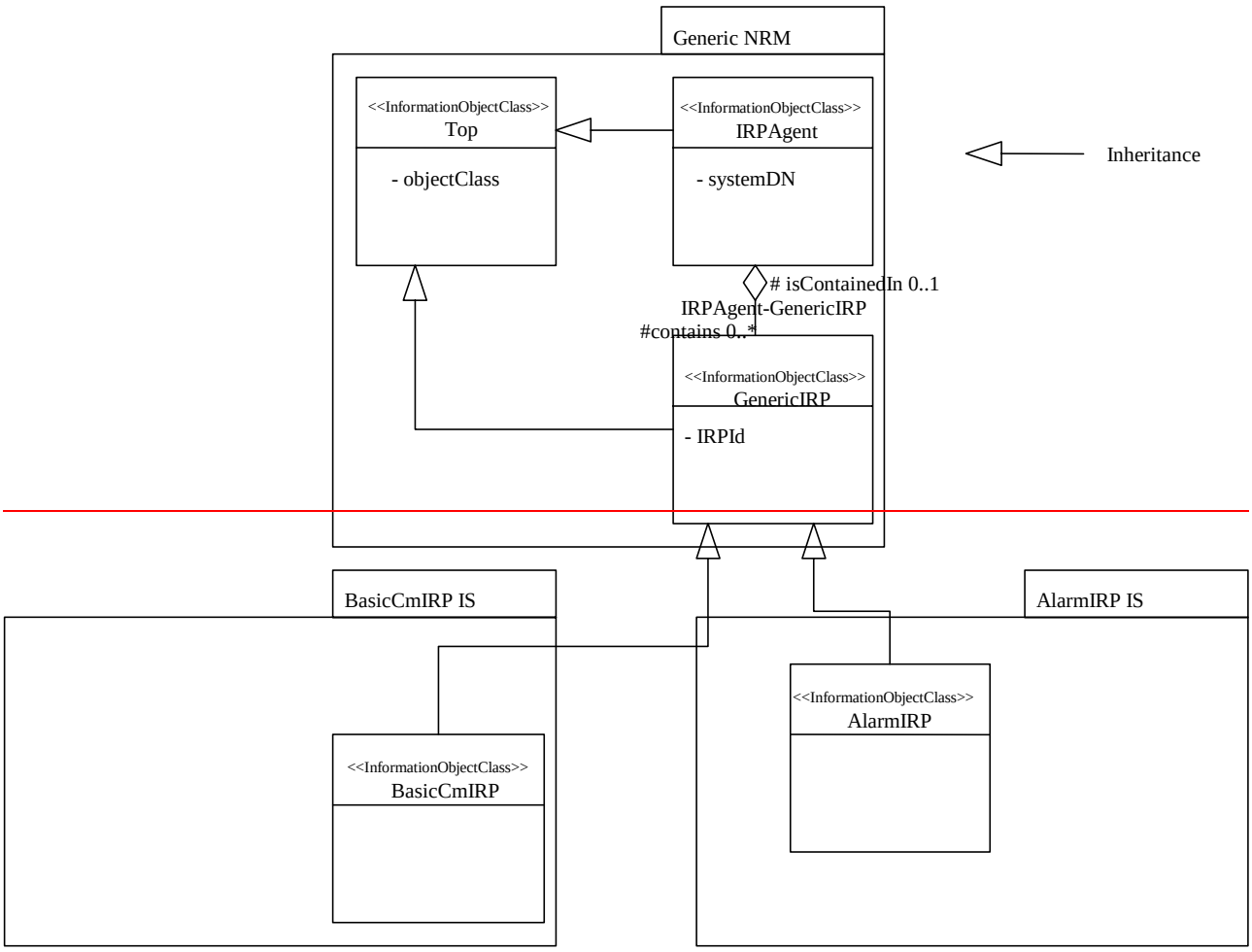


Figure D.1: Example of possible packages together with information object classes and their inter-relationships

The following aspects are illustrated in figure D.1:

1) Information object classes that are common to all Network Resources Models / some Information Services are captured in the GenericNRM package : Top, IRPAgent, GenericIRP, together with their attributes and relationships;

2) The information object class BasicCmIRP is defined in the BasicCmIRP IS package. As illustrated in the previous figure, this package imports the GenericNRM package;

3) The information object class AlarmIRP is defined in the AlarmIRP IS package. As illustrated in the previous figure, this package imports the GenericNRM package;

4) As a consequence, every information object class can inherit from the class Top, either directly or indirectly;

5) The IRPAgent class is defined in the GenericNRM;

6) A GenericIRP information object class is defined in the Generic NRM. It represents an abstraction of all the IRPs such as, e.g. BasicCmIRP or AlarmIRP. A containment relationship between IRPAgent and GenericIRP is defined;

7) Both the information object classes BasicCmIRP and AlarmIRP (defined in different Information Services) inherit from GenericIRP. As a first consequence, they inherit the attributes IRPId and IRPVersion (from GenericIRP) and objectClass (from Top). As a second consequence, both BasicCmIRP and AlarmIRP are contained in IRPAgent.

Annex E (informative): General rules for Solution Sets (SSs)

E.1 Introduction

The intent of this annex is twofold. The first intent is for 3GPP internal use to document how a 3GPP SS is produced and what it shall contain. The second intent with the annex is to give the reader of an Information Service (IS) or a SS a better understanding on how to interpret the IS or SS specifications.

E.2 Solution Set (SS) versioning

For further study.

E.3 Referenced Information Service (IS) specification

A sentence shall be included in the clause "Scope" of all SS specifications. The sentence shall read as follows:

"This Solution Set specification is related to Z"

where Z is the 3GPP Information Service specification number including the version, such as "TS 32.111-2 V4.1.X" for the case of Alarm Integration Reference Point: Information Service.

NOTE: that "X", rather than the actual digit, is actually used in the sentence. This is because the value of X is not relevant for the reference purpose since different values of X identify different 3GPP published specifications that reflect only minor editorial changes.

Annex F (normative): Rules for CORBA Solution Sets (SSs)

F.1 Introduction

The intent of this annex is threefold. The first intent is for 3GPP internal use to document how a 3GPP CORBA SS is produced and how it is structured. The second intent with the annex is to give the reader or implementer of a CORBA SS a better understanding on how to interpret the CORBA SS document. The last and maybe most important intent is to put requirement on an implementer of a CORBA SS.

It can be noted that it is expected that this annex is to be extended in later versions of the present document.

F.2 Rules for specification of CORBA Solution Sets (SSs)

F.2.1 Introduction

This clause identifies rules for specification of CORBA SSs. This clause is mainly for 3GPP internal use. It is only for information for the implementer of a CORBA SS.

F.2.2 Pragma prefix

All IDL code shall define the pragma prefix using the following statement:

#pragma prefix "3gppsa5.org"

F.3 Implementation aspects of CORBA Solution Sets (SSs)

F.3.1 Introduction

This clause identifies rules for the implementation of CORBA SSs. This clause is normative for the implementer of a CORBA SS.

F.3.2 IRPAgent behaviour on incoming optional method

The IRPAgent, claiming compliance to a particular SS version of a particular IRP such as the Alarm IRP, shall implement all mandatory and all optional methods. Each method implementation shall have a signature specifying all mandatory and all optional parameters.

- If the IRPAgent does not support a particular optional method, it shall throw the `OperationNotSupported` exception when the IRPManager invokes that method.
- If the IRPAgent have not implemented a particular method (because it is compiled with an IDL version that does not define the method), the CORBA ORB of the IRPAgent shall throw a system exception if the IRPManager invokes that method.

In all the above cases when an exception is thrown, the IRPAgent shall restore its state before the method invocation.

F.3.3 IRPAgent Behaviour on incoming optional parameter of operation

An IRPAgent must implement all optional parameters, as well as mandatory parameters, in all methods.

If the IRPAgent supports the implemented method but does not support its (one or more) optional input parameters, upon method invocation, the IRPAgent shall check if those parameters carry "no information" or absence semantics (defined later in sub-clause "Encoding rule for absence semantics"). If the check is negative, the IRPAgent shall throw the `ParameterNotSupported` exception with a string carrying the name of the unsupported optional parameter.

F.3.4 IRPAgent Behaviour on outgoing attributes of Notification

CORBA SS uses OMG defined structured event to carry notification. The structured event is partitioned into header and body.

The absence semantics of attribute in the header is realised by a string of zero length.

The body consists of one or more name-value pair attributes. The absence semantics of these attributes is realised by their absence.

For optional sub-attributes of an attribute carried by the name-value pair, their absence semantics is realised by the encoding rule of "absence semantics". See sub-clause E.3.5, "Encoding rule of absence semantics".

F.3.5 Encoding rule of absence semantics

The operation parameters are mapped to method parameters of CORBA SS. The absence semantics for an operation (input and output) parameter is method parameter type dependent.

- For a string type, if the parameter is specified as a string type, the absence semantics is a string of zero length. If the parameter is specified as a union structure (preferred), the absence semantics is conveyed via a FALSE Boolean value switch.
- For an integer type, if the parameter is specified as a signed, unsigned, long, etc type, the absence semantics is the highest possible positive number. If the parameter is specified as a union structure (preferred), the absence semantics is conveyed via a FALSE Boolean value switch.
- For a boxed valueType (supported by CORBA 2.3), it is the null value.

The notification parameters are mapped to attributes of the CORBA Structured Events. The absence semantics for a notification parameter is attribute position (within the Structured Event) dependent.

- For the fixed header of the Structured Event header, the absence semantics is realised by a string of zero length.
- For the filterable body fields of the Structured Event body, the absence semantics is realised by the absence of the corresponding attribute.

F.4 IDL file name rule

CORBA IDL uses "#include "X"" statement where X is a name of a file containing IDL statements. In the CORBA Solution Set, IDL statements are specified.

The rule defined here specifies

- the IDL statements that shall belong to one file; and
- the name of the file.

Rule: In the Annex where IDL statements are defined, use a special marker to indicate that a set of IDL statements shall be contained in one file. The name of the file shall be the name of the first IDL module of that set (of IDL statements). Multiple markers can be used.

Annex G (normative): IRP IS UML Modelling Repertoire

G.1 Introduction

3GPP SA5 has chosen UML to capture systems behaviour in the IRP IS context.

UML provides a rich set of concepts, notations and model elements to model distributive systems. Usage of all UML notations and model elements is not necessary for the purpose of IRP IS specifications. This annex documents the necessary and sufficient set of UML notations and model elements, including the ones built by the UML extension mechanism <<stereotype>>, for use by 3GPP IRP IS authors. Collectively, this set of notations and model elements is called the 3GPP IRP IS modelling repertoire.

The selection of the UML notations and model elements in this repertoire is based on the needs of the existing 3GPP IRP IS specifications. Future IRP IS releases may require the use of additional UML notations or model elements.

IRP IS specifications shall employ the UML notation and model elements of this repertoire and may also employ other UML notation and model elements considered necessary. However, before any other UML notation and model elements may be employed in an approved 3GPP IRP specification, the other notation and model elements should be agreed for inclusion first in this repertoire.

All quotes are from [15].

Capitalized words are defined by various 3GPP IRP IS specifications or the reference [15].

G.2 Requirements

IRPAgent can be characterized by several different but related models. The models can be exterior or interior to the IRPAgent. Exterior models are use case models and interior models are object models.

Current version of this Annex focuses on the interior model aspects of IRPAgents.

The notation elements captured in this repertoire shall be used to model all aspects of NRM IRP IS (such as GERAN NRM IRP: IS) and (protocol) IRP (such as Alarm IRP: IS).

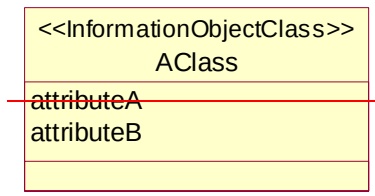
G.3 Model Elements and Notations

G.3.1 Basic Model Elements

UML defined a number of basic Model Elements. This subclause lists the selected subset for use in the repertoire. The semantics of the selected ones are defined in [15].

attribute (Subclause 3.25 of [15]).

This sample shows two attributes, listed as strings in the attribute compartment of the class AClass.

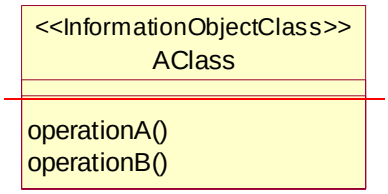


aggregation (Subclause 3.43.2.5 of [15]).

~~This sample shows a hollow diamond attached to the end of a path to indicate aggregation. The diamond is attached to the class that is the aggregate.~~

~~operation (Subclause 3.26 of [15]).~~

~~This sample shows two operations, shown as strings in the operation compartment of class AClass, that the instance of AClass may be requested to perform. The operation has a name, e.g. operationA and a list of arguments (not shown).~~



~~association, association name (Subclause 3.41 of [15]).~~

~~This sample shows a binary association between exactly two model elements. The association can include the possibility relating a model element to itself. This sample shows a bi-directional association in that one model element is aware of the other. Association can be unidirectional (shown with an open arrow at one association end) in that only the source model element is aware of the target model element and not vice versa.~~

~~realization relationship (Subclause 2.5.2.1 of [15]).~~

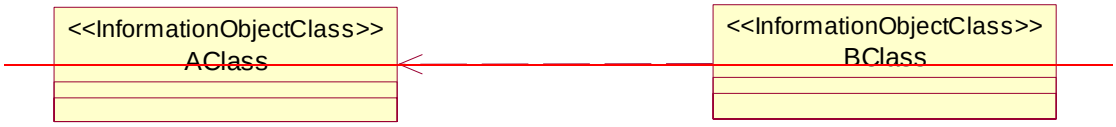
~~This sample shows the realization relationship between a AlarmIRPNotification_1 (the supplier) and a model element, IRPManager, that implements it.~~

~~generalization relationship (Subclause 3.50 of [15]).~~

~~This sample shows a generalization relationship between a more general element (the IRPAgent) and a more specific element (the IRPAgent_vendor_A) that is fully consistent with the first element and that adds additional information.~~

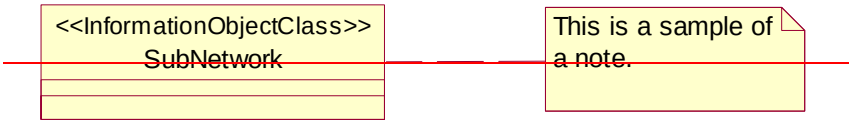
~~dependency relationship (subclause 3.51 of [15]).~~

~~This sample shows that BClass instances have a semantic relationship with AClass instances. It indicates a situation in which a change to the target element will require a change to the source element in the dependency.~~



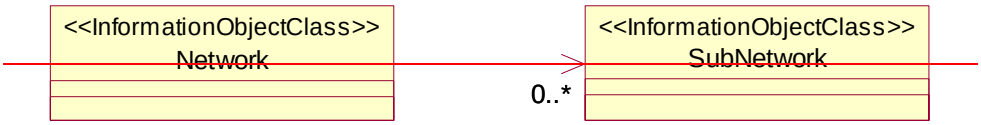
note (Subclause 3.11 of [15])

This sample shows a note, as a rectangle with a "bent corner" in the upper right corner. The note contains arbitrary text. It appears on a particular diagram and may be attached to zero or more modelling elements by dashed lines.



Multiplicity, a.k.a. cardinality (Subclause 3.44 of [15]).

This sample shows a multiplicity attached to the end of an association path. The meaning of this multiplicity is that one Network instance is associated with zero, one or more SubNetwork instances.



rolename (Subclause 3.43.2.6 of [15]).

This sample shows a Person (say instance John) is associated with a Company (say instance XYZ). We navigate the association by using the opposite association-end such as John.theCompany ="XYZ". Use noun for the rolename.

G.3.2 Stereotype

This sub-clause defines all allowable stereotypes that are summarized in the following table. Except <<Interface>>, <<Type>> and <<use>> (which are defined in [15]), all other stereotypes are extensions specifically designed for use in IRP IS specifications.

Table G.1: Stereotypes

Stereotype	Base Class	Affected Metamodel Elements
Interface	Class	
Type	Class	
ProxyClass	Class	
Archetype	Classifier (subclause 2.5.2.10 of [15])	
InformationObjectClass	Classifier	
use	Association	
may use	Association	
may realize	Association	
emits	Association	
names	Composition	--
opt (alternatively «optional»)	ModelElement	Attribute, Parameter, and Operation
%	3GPPVisibilityKind	--

G.3.2.1 <<Interface>>

Subclause 2.5.2.25 of [15]:

"An interface is a named set of operations that characterize the behaviour of an element. In the metamodel, an Interface contains a set of Operations that together define a service offered by a Classifier realizing the Interface. A Classifier may offer several services, which means that it may realize several Interfaces, and several Classifiers may realize the same Interface.

…

Interfaces may not have Attributes, Associations, or Methods. An Interface may participate in an Association provided the Interface cannot see the Association; that is, a Classifier (other than an Interface) may have an Association to an Interface that is navigable from the Classifier but not from the Interface."

Subclause 2.5.4.6 of [15]: "The purpose of an interface is to collect a set of operations that constitute a coherent service offered by classifiers. Interfaces provided a way to partition and characterize groups of operations. An interface is only a collection of operations with a name. It cannot be directly instantiated. Instantiable classifiers, such as class or use case, may use interfaces for specifying different services offered by their instances. Several classifiers may realize the same interface. All of them must contain at least the operations matching those contained in the interface. The specification of an operation contains the signature of the operation (i.e. its name, the types of the parameters and the return type). An interface does not imply any internal structure of the realizing classifier. For example, it does not include which algorithm to use for realizing an operation. An operation may, however, include a specification of the effects [e.g. with pre and post conditions] of its invocation."

G.3.2.1.1 Sample

This sample shows an AlarmIRPOperations_1 <<Interface>> that has two operations. The operation visibility is public (see definition of public visibility applicable to operation in subclause "visibility"). The input and output parameters of the operations are hidden (i.e. not shown). The AlarmIRP has a unidirectional mandatory realisation relationship with the <<interface>>.

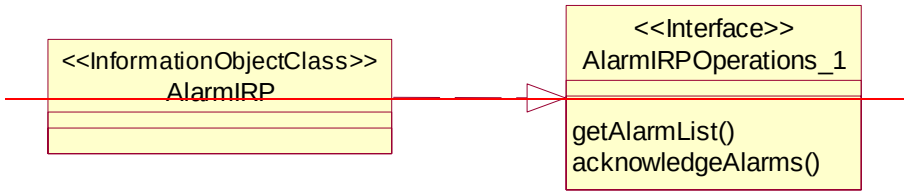


Figure : <<Interface>> Notation

G.3.2.2 <<Type>>

Subclause 3.28 of [15]: "[A Type is] a domain of objects together with the operations applicable to the objects, without defining the physical implementation of those objects. A Type may not contain any methods, maintain its own thread of control, or be nested. However, it may have Attributes and Associations. The Associations of a Type are defined solely for the purpose of specifying the behaviour of the Type's operations and do not represent the implementation of state data".

G.3.2.2.1 Sample

This sample shows the NotificationIRPNotification <<Type>> that specifies the five parameters (the notification header of Notification IRP). The AlarmIRPNotification_2 <<Interface>> depends (see the dependency relationship, a dashed open arrow line) on this <<Type>> for the construction of the notification emitted via the operation notifyChangedAlarm(). The visibility of attributes and operation in the example is public.

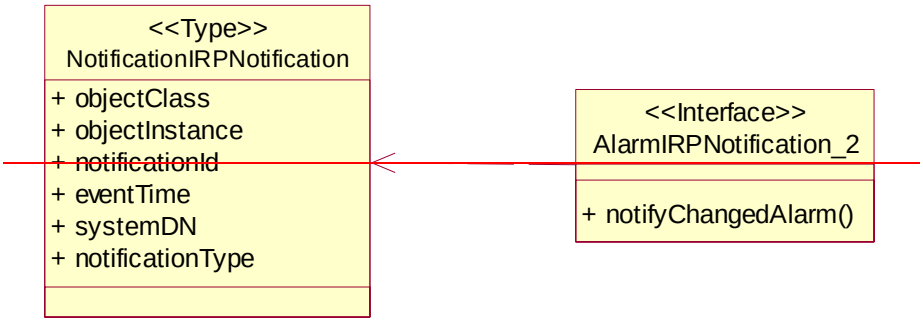


Figure : <<Type>> Notation

G.3.2.3 <<ProxyClass>>

It is a form or template representing a number of <<InformationObjectClass>>. It encapsulates attributes, links, methods (or operations), and interactions that are present in the represented <<InformationObjectClass>>.

The semantics of a <<ProxyClass>> is that all behaviour of the <<ProxyClass>> are present in the represented <<InformationObjectClass>>. Since this class is simply a representation of other classes, this class cannot define its own behaviour other than those already defined by the represented <<InformationObjectClass>>.

A particular <<InformationObjectClass>> can be represented by zero, one or more <<ProxyClass>> or <<Archetype>>. For example, the ManagedElement <<InformationObjectClass>> can have MonitoredEntity <<ProxyClass>> and ManagedEntity <<ProxyClass>>.

The attributes of the <<proxyClass>> are accessible by the source entity that has an association with the <<ProxyClass>>.

G.3.2.3.1 Sample

This shows a <<ProxyClass>> named MonitoredEntity. It represents all NRM <<InformationObjectClass>> (e.g. GgsnFunction <<InformationObjectClass>>) whose instances are being monitored for alarm conditions. The MonitoredEntity plays the role of theMonitoredEntity.

Note that <<MonitoredEntity>> does not define attributeA. The attributeA is already defined by all <<InformationObjectClass>> represented by the <<MonitoredEntity>>, i.e. ClassA and ClassB.

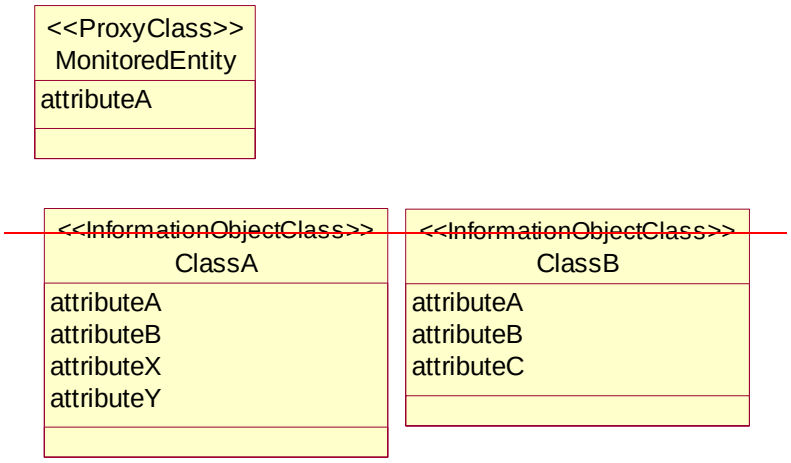


Figure : <<ProxyClass>>

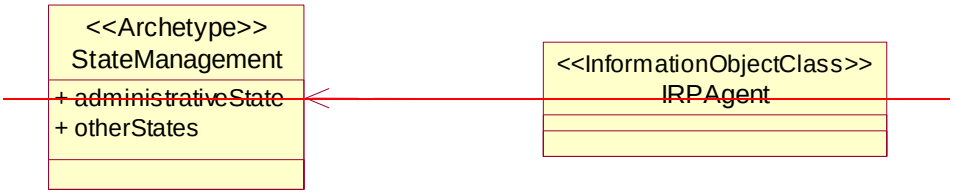
G.3.2.4 <<Archetype>>

It is a form or template representing a number of <<InformationObjectClass>>. It encapsulates attributes, links, operations, and interactions that are typical of the represented <<InformationObjectClass>>.

The semantics of an <<archetype>> is that all attributes, links operations and interactions encapsulated by the <<archetype>> may or may not be present in the represented <<InformationObjectClass>>. The <<Archetype>> represents a placeholder class that is most useful in technology neutral analysis models that will require further specification and/or mapping within a more complete construction model.

G.3.2.4.1 Sample

This shows a <<Archetype>> named StateManagement. It also shows a <<InformationObjectClass>> IRPAgent that depends on this StateManagement. Note that the StateManagement has defined a number of attributes. The classes that depend on this StateManagement may or may not use all of the StateManagement attributes. In other words, at least one of the attributes of StateManagement is present in the IRPAgent. The precise set of StateManagement attributes used by the IRPAgent is specified in the IRPAgent specification.



G.3.2.5 <<InformationObjectClass>>

It is the descriptor for a set of network resources and network management capabilities. Each <<InformationObjectClass>> represents a set of instances with similar structure, behaviour and relationships.

This <<InformationObjectClass>> and other information classes such as <<interface>> are mapped into technology specific model elements such as GDMO Managed Object Class for CMIP technology. The mapping of IS modelling constructs to technology specific modelling constructs are captured in the corresponding IRP Solution Set specifications.

The name of a <<InformationObjectClass>> has scope within the 3GPP IRP IS document in which it is specified and the name must be unique among all <<InformationObjectClass>> names within that 3GPP IRP IS document. The IRP IS document name is considered in the similar way as the UML Package name.

The <<InformationObjectClass>> is identical to UML *class* except that it does not include/define methods or operations.

Subclause 3.22.1 of [15]: "A *class* represents a concept within the system being modelled. Classes have data structure and behaviour and relationships to other elements."

G.3.2.5.1 Sample

This sample shows an AlarmList <<InformationObjectClass>>.

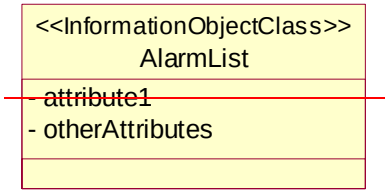


Figure : <<InformationObjectClass>>> Notation

G.3.2.6 <<use>> and <<may use>>

The <<use>> and <<may use>> are unidirectional associations. The target must be an <<interface>>. The <<use>> states that the source class must have the capability to use the target <<interface>> in that it can invoke the operations defined by the <<interface>>. Support of the capability by the source entity is mandatory. The <<may use>> states that the source class may have the capability to use the target <<interface>> in that it may invoke the operations defined by the <<interface>>. Support of the capability by the source entity is optional.

The operations defined by the <<interface>> are visible across the itf-N.

G.3.2.6.1 Sample

This shows that the NotificationIRPAgent shall use the notifyNewAlarm and otherNotifications of AlarmIRPNotification_1 and may use the notifyChangedAlarm of AlarmIRPNotification_2.

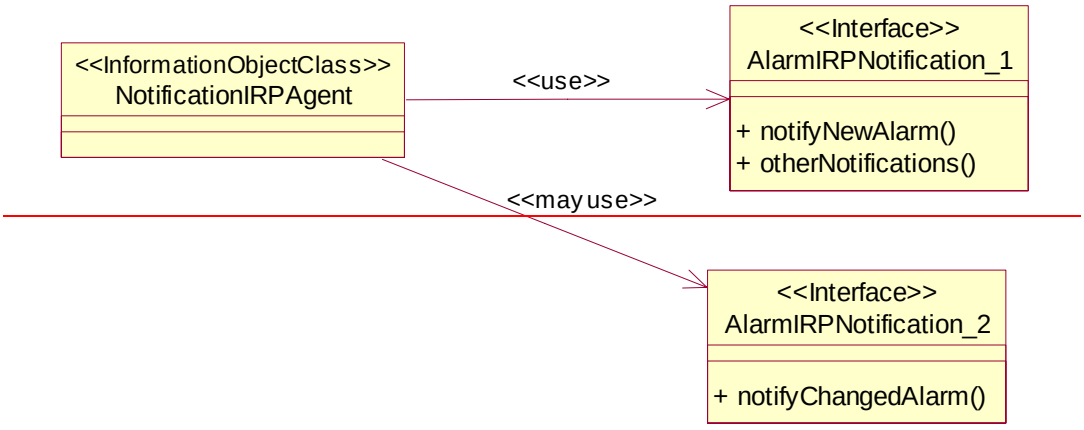


Figure : <<use>> and <<may use>> Notation

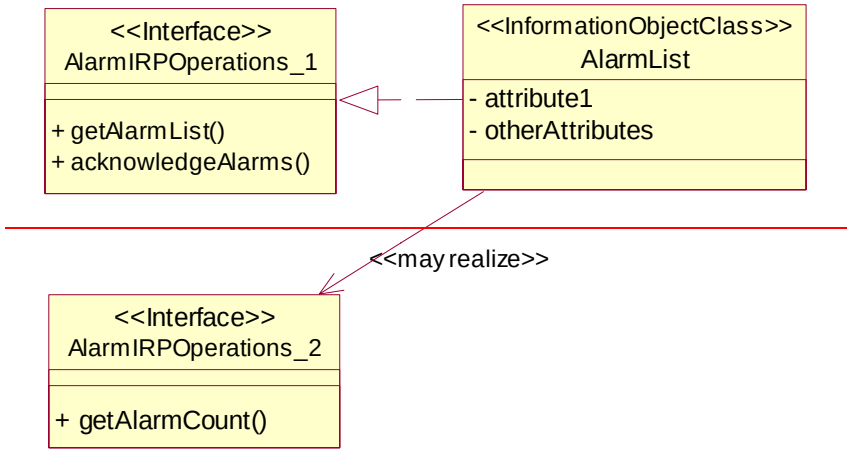
G.3.2.7 Relationship realize and <<may realize>>

The relationship realize and <<may realize>> are unidirectional association. The target must be an <<interface>>. The relationship "realize" shows that the source entity must realize the operations defined by the target <<interface>>. Realization of operations by the source entity is mandatory. The <<may realize>> shows the source entity may realize the operations defined by the target <<interface>>. Realization of the <<interface>> by the source entity is optional.

The operations defined by <<interface>> are visible across the itf-N.

G.3.2.7.1 Sample

This shows that the AlarmList shall realize (or support, implement) the two operations of AlarmIRPOperations_1 and may realize the operation of AlarmIRPOperations_2.



~~Figure : Relationship realize and <<may realize>> Notations~~

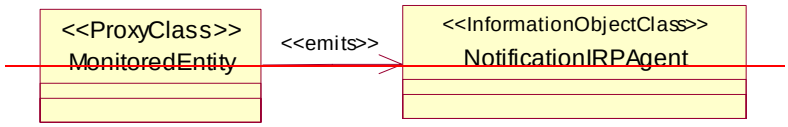
~~G.3.2.8 <<emits>>~~

~~This is a unidirectional association. The source sends information to target. In the case that the target is NotificationIRPAgent, the information will then carry the semantics of 3GPP notification (e.g. notifyObjectCreation, notifyNewAlarm) such that the target NotificationIRPAgent can construct the relevant 3GPP notification for reception by the NotificationIRPManager.~~

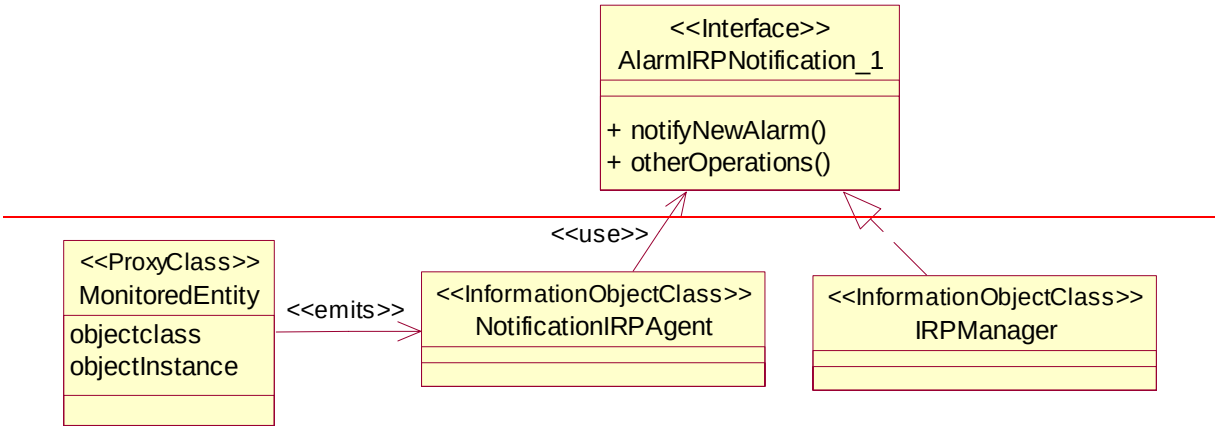
~~The visibility of the information passed by <<emits>> is always "IRPAgent Internal" (see subclause on "Visibility").~~

~~G.3.2.8.1 Sample~~

~~This shows the MonitoredEntity (e.g. a GgsnFunction instance) emits notifications that are received by the NotificationIRPAgent. The emission is not visible across the itf-N.~~



~~Figure : <<emits>> Notation~~



~~Figure : <<use>>, <<emits>> and realize relationship Notation~~

G.3.2.9 <<names>>

It specifies a unidirectional composition. The target instance is uniquely identifiable, within the namespace of the source entity, among all other targeted instances of the same target classifier and among other targeted instances of other classifiers that has the same «names» composition with the source.

Composition used as the act of name containment provides a semantic of a whole-part relationship between the domain and the named elements that are contained, even if only by name. From the management perspective access to the part is through the whole. Multiplicity shall be indicated on both ends of the relationship.

A target cannot have multiple <<names>> with multiple sources, i.e. a target cannot participate in or belong to multiple namespaces.

By convention, the name of the attribute in the target model element to hold part of the unique identification shall be formed by the name of the target class concatenated with "Id". There are two presentation options for the unique identification attribute of the class being named.

- The use of the role qualifier allows the unique identification attribute to be attached to the target end of the «names» association (see figure G.9).
- The unique identification attribute may be indicated as a normal attribute within the class attribute compartment.

G.3.2.9.1 Sample

This shows that all instances of ManagedFunction are uniquely identifiable within the ManagedElement namespace. Note the use of the label supports in specifications is optional.

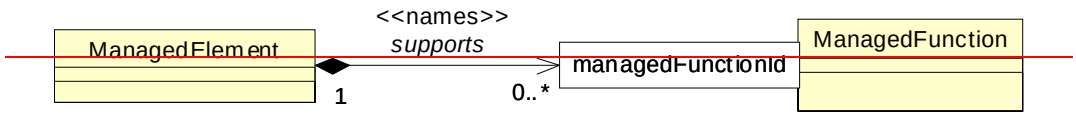


Figure : <<names>> Notation, Composition and explicit Qualifier

G.3.2.10 «opt»

The «opt» enables the indication of optionality of attributes, parameters and operations (respectively) within the UML diagrams within TS 32 series documents. The semantics of optionality are clearly defined within subclause 10.6 of this document.

In the absence of the «opt» stereotype, the attribute, parameter, or operation in question is mandatory.

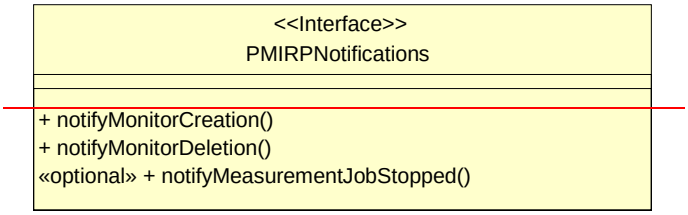


Figure (a): Example of the use of optionality indicator for operations

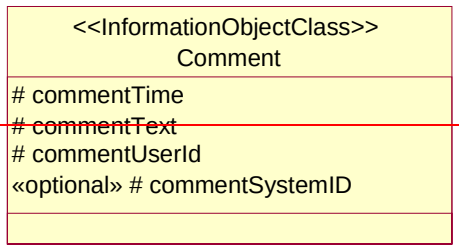


Figure (b): Example of the use of optionality indicator for attributes

G.3.3 Visibility

It specifies the accessibility of the operation and attribute. There are three types of visibility, i.e. private, public and IRPAgent Internal.

Table G.2: Private Visibility (notation "-")

Operation	NA
Attribute	It indicates that the attribute is not accessible by other entities, e.g. the IRPManager or other entities not holding the subject attribute.

Table G.3: Public Visibility (notation "+")(default)

Operation	It indicates that the operation is visible across the itf-N, e.g. the IRPManager can invoke the operation across the itf-N interface.
Attribute	It indicates that the attribute is accessible across the itf-N, i.e. the IRPManager can invoke an operation to read the attribute and to write to this attribute if the attribute is so qualified. The read or write operation must be directly invoked against the entity holding the subject attribute or against the CM IRP Agent.

Table G.4: IRPAgent Internal Visibility (notation "%")

Operation	It indicates that the operation is not visible across the itf-N, i.e. the IRPManager cannot invoke the operation. However, other entities can invoke the operation. (Note: no Release 5 operations are of this kind.)
Attribute	It indicates that the attribute is not directly accessible across the itf-N, i.e. the IRPManager cannot read/write this attribute. However, other entities can read/write this attribute.

G.3.3.1 Samples

This sample shows four attributes whose visibility are private, public (default notation), public and IRPAgent Internal. It is recommended that within a Class symbol, the use of default notation or not for public visibility should be consistent, i.e. all "publicly visible" attributes shall be shown with the "+" sign or without the "+" sign (default notation).

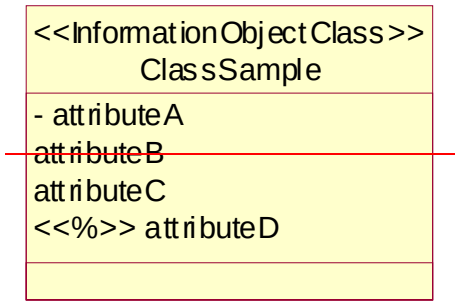
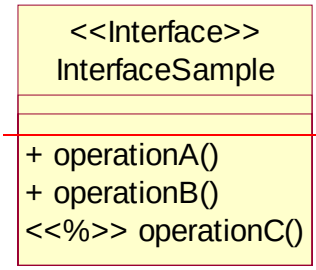


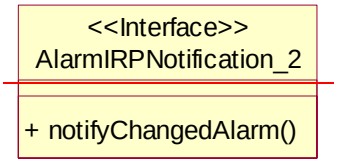
Figure : Visibility of attributes

This sample shows three operations. Two of these operations are accessible by the IRPManager via the itf-N. It is recommended that within a Class symbol, the use of default notation or not for public visibility should be consistent, i.e. all "publicly visible" operation shall be shown with the "+" sign or without the "+" sign (default notation).



~~**Figure : Visibility of operations**~~

~~This sample shows one notification whose visibility is public using the non-default public visibility notation. These notifications are accessible by the IRPManager via the itf-N.~~



~~**Figure : Visibility of notification**~~

End of Deletion of Annex C, D, E, F & G

End of Document