INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Z.360

(05/97)

SERIES Z: PROGRAMMING LANGUAGES

Man-machine language – Specification of the man-machine interface

Graphic GDMO: A graphic notation for the Guidelines for the Definition of Managed Objects

ITU-T Recommendation Z.360

(Previously CCITT Recommendation)

ITU-T Z-SERIES RECOMMENDATIONS

PROGRAMMING LANGUAGES

Specification and Description Language (SDL)	Z.100–Z.109
Applicability of formal Description Techniques	Z.110–Z.119
Message Sequence Chart	Z.120–Z.199
CCITT High Level Language (CHILL)	Z.200–Z.299
MAN-MACHINE LANGUAGE	Z.300–Z.499
General principles	Z.300–Z.309
Basic syntax and dialogue procedures	Z.310–Z.319
Extended MML for visual display terminals	Z.320–Z.329
Specification of the man-machine interface	**Z.330–Z.399**
Miscellaneous	Z.400–Z.599

For further details, please refer to ITU-T List of Recommendations.

ITU-T RECOMMENDATION Z.360

GRAPHIC GDMO: A GRAPHIC NOTATION FOR THE GUIDELINES FOR THE DEFINITION OF MANAGED OBJECTS

Summary

This Recommendation prescribes a graphic notation to provide an overview of GDMO/GRM specifications.

GDMO, Guidelines for the Definition of Managed Objects, is a formalism defined in Recommendation X.722 to define managed objects in Telecommunications Management Network (TMN). Attributes for representing relationships are defined in Recommendation X.732. The GDMO formalism is extended with a general relationship notation in Recommendation X.725 (General Relationship Model). The overall object structure is defined by GDMO/GRM, while the syntax of the data is defined using Abstract Syntax Notation One (ASN.1), Recommendations X.208 or X.680. The three languages are widely used by many ITU Study Groups for the specification of managed objects for many application areas. The languages are purely alphanumeric and the alphanumeric specifications using GDMO/GRM/ASN.1 are hard to overview.

Current Recommendations using GDMO/GRM use various kinds of informal illustrations to provide overviews of different aspects of the specifications. However, they frequently use the same symbol for different notions, different symbols for the same notion, and can sometimes be misleading as to the contents of the specifications. Therefore, a standardized graphic notation was felt needed. The graphic notation shall be true to the notions of the alphanumeric notations.

This Recommendation is based on:

- an approved Requirement document, found in Appendix II;
- evaluation of some existing graphic notations, which did not meet the requirements;
- trial use on several Recommendations and other application areas.

Source

ITU-T Recommendation Z.360 was prepared by ITU-T Study Group 10 (1997-2000) and was approved under the WTSC Resolution No. 1 procedure on the 6th of May 1997.

Keywords

Attributes; General Relationship Model; Guidelines for the Definition of Managed Objects; Managed Object Class; Relationship Class.

FOREWORD

ITU (International Telecommunication Union) is the United Nations Specialized Agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of the ITU. The ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Conference (WTSC), which meets every four years, establishes the topics for study by the ITU-T Study Groups which, in their turn, produce Recommendations on these topics.

The approval of Recommendations by the Members of the ITU-T is covered by the procedure laid down in WTSC Resolution No. 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

INTELLECTUAL PROPERTY RIGHTS

The ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. The ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, the ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementors are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database.

CONTENTS

Page

Recommendation Z.360

GRAPHIC GDMO: A GRAPHIC NOTATION FOR THE GUIDELINES FOR THE DEFINITION OF MANAGED OBJECTS

(Geneva, 1997)

1 Introduction

1.1 Scope

This Recommendation provides a graphic notation for a subset of the information provided in the templates defined in Recommendation X.722, Information technology – Open Systems Interconnection – Structure of management information: Guidelines for the definition of managed objects.

The purpose of the graphic notation is to provide an overview of TMN specifications for TMN specifiers, implementors and users. The graphic notation provides an overview of what definitions exist and the relationships between these definitions.

To accomplish this overview, not all information provided in the templates has to be presented in the graphic notation. Therefore, the graphic notation covers only a subset of the information found in and required by the templates.

If inconsistencies between Alphanumeric GDMO/GRM and Graphic GDMO specifications are discovered, the alphanumeric specification is normative.

The graphic notation allows both detailed and compact depictions of specifications. The compact notation is expected to be the most useful one for ordinary users. The detailed notation is a tool for explaining the detailed relationships between the statements of the specifications and to show the relationships to the alphanumeric notation.

Typically, a graph may depict all information in one Recommendation. However, the specifier is free to include and exclude what he finds convenient, graphs can overlap, and there is no Recommendation on what shall be included in one graph or not.

A Graphic GDMO graph should be provided with a title and text indicating how this graph is delimited relative to the alphanumeric specifications to which it refers.

Clause 2 introduces the graphic notation and concludes with a made-up example (see Figure 4) using the most compact form of the notation.

Clause 3 provides a real world example from the area of Access Control.

Clauses 4 and 5 provide a more detailed explanation of the example in Clause 2, Figure 4, and expansion from the compact notation to show all the details.

Clause 6 provides an extension of the graphic notation to the Relationship Class template defined in Recommendation X.725.

Appendix I extends the graphic notation for classes with a separate, however, similar graphic notation for depicting instances.

Appendix II contains the requirements which Graphic GDMO was developed to meet.

1.2 Terminology

This Recommendation defines the following terms:

1.2.1 graphic GDMO/GRM: This is used as a full name of the graphic notation for GDMO/GRM. The short name for Graphic GDMO/GRM is Graphic GDMO.

1.2.2 **graphic GDMO/GRM specifications**: Specifications using Graphic GDMO/GRM. These specifications are synonymously called graphs or drawings.

1.2.3 **template**: A complete permissible syntax module in Alphanumeric GDMO/GRM.

1.2.4 **statement**: An instance of a GDMO/GRM template together with filled in application specific labels and parameters. The labels may refer to other GDMO/GRM statements or non-GDMO/GRM statements (e.g. in ASN.1).

1.2.5 **subtemplate**: A part of a complete GDMO/GRM template.

1.2.6 **substatement**: A part of a complete GDMO/GRM statement.

1.2.7 **symbol**: An item in Graphic GDMO/GRM which corresponds to a template.

1.2.8 **connective**: An item in Graphic GDMO/GRM which corresponds to a reference between templates or a series of templates and references.

2 Symbols

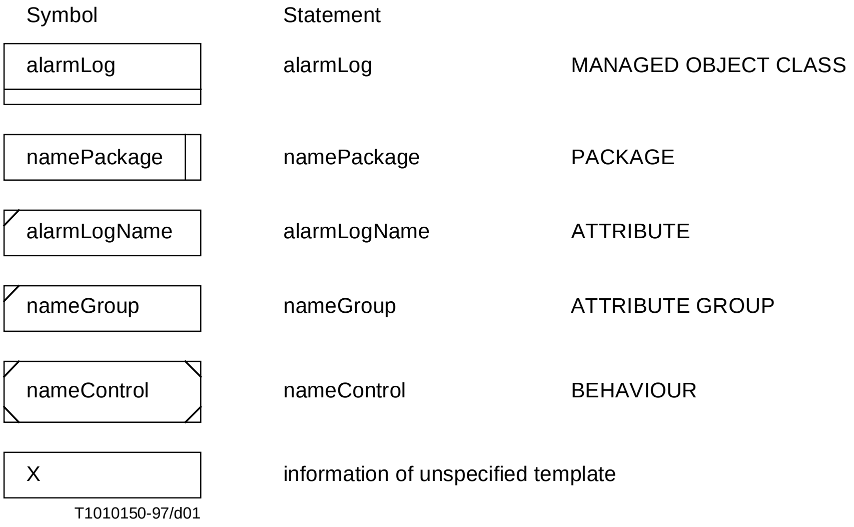


Figure 1/Z.360 – Template statement symbols

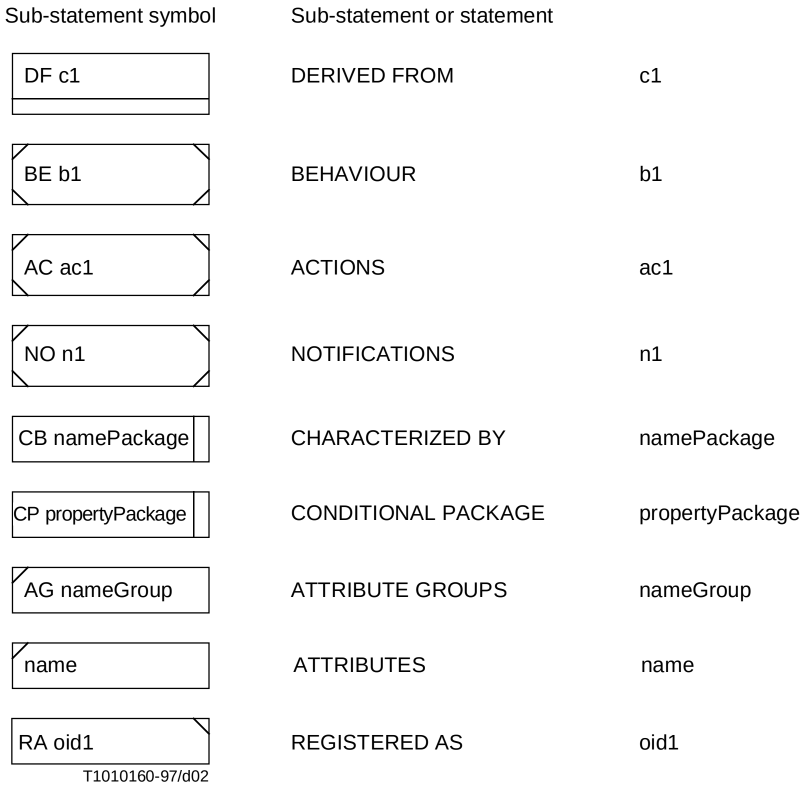
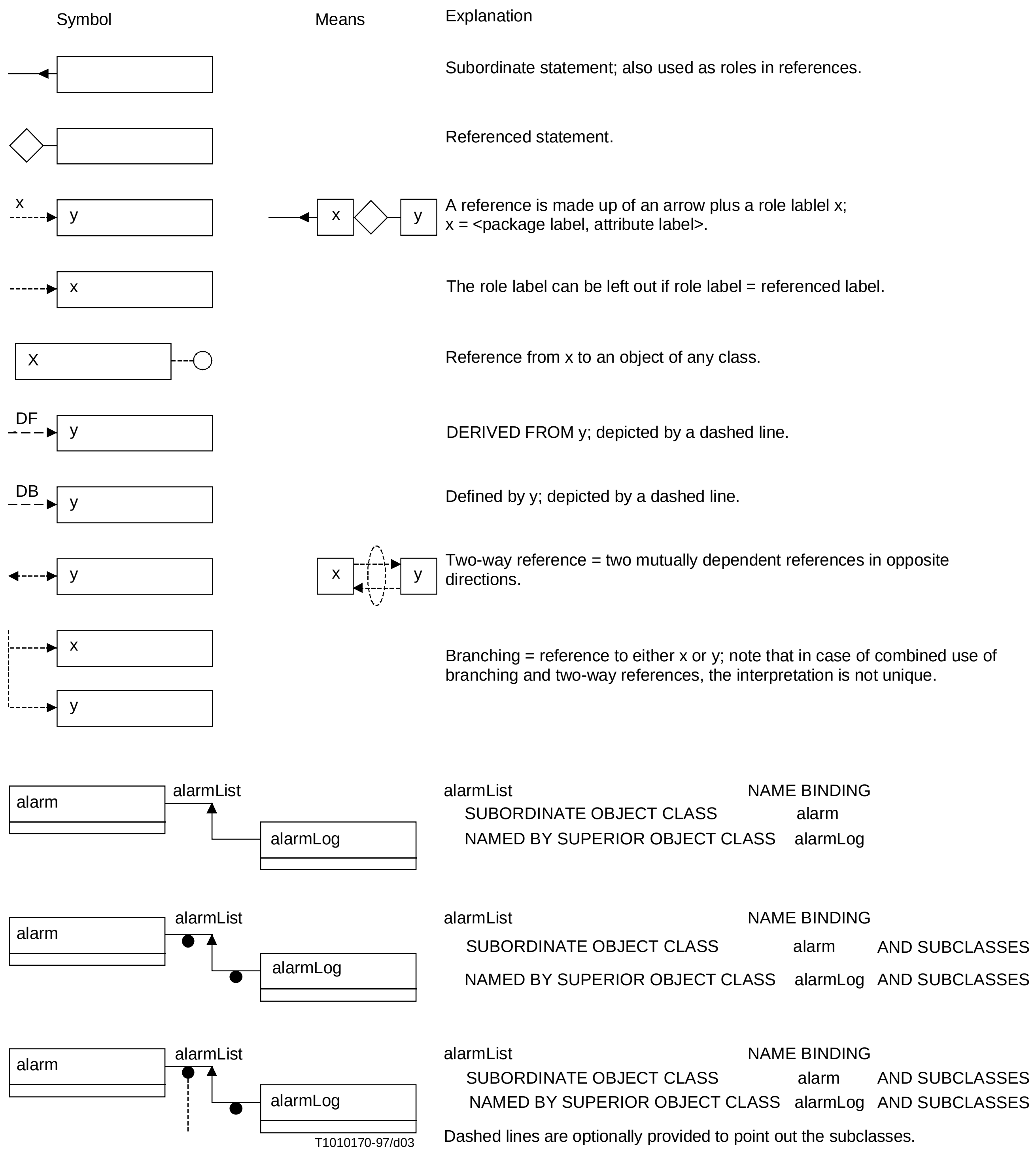


Figure 2/Z.360 – Sub-statement symbols



Subordinate statement; also used as roles in references.

Referenced statement.

A reference is made up of an arrow plus a role lablel x;
x = <package label, attribute label>.

The role label can be left out if role label = referenced label.

Reference from x to an object of any class.

DERIVED FROM y; depicted by a dashed line.

Defined by y; depicted by a dashed line.

Two-way reference = two mutually dependent references in opposite directions.

Branching = reference to either x or y; note that in case of combined use of branching and two-way references, the interpretation is not unique.

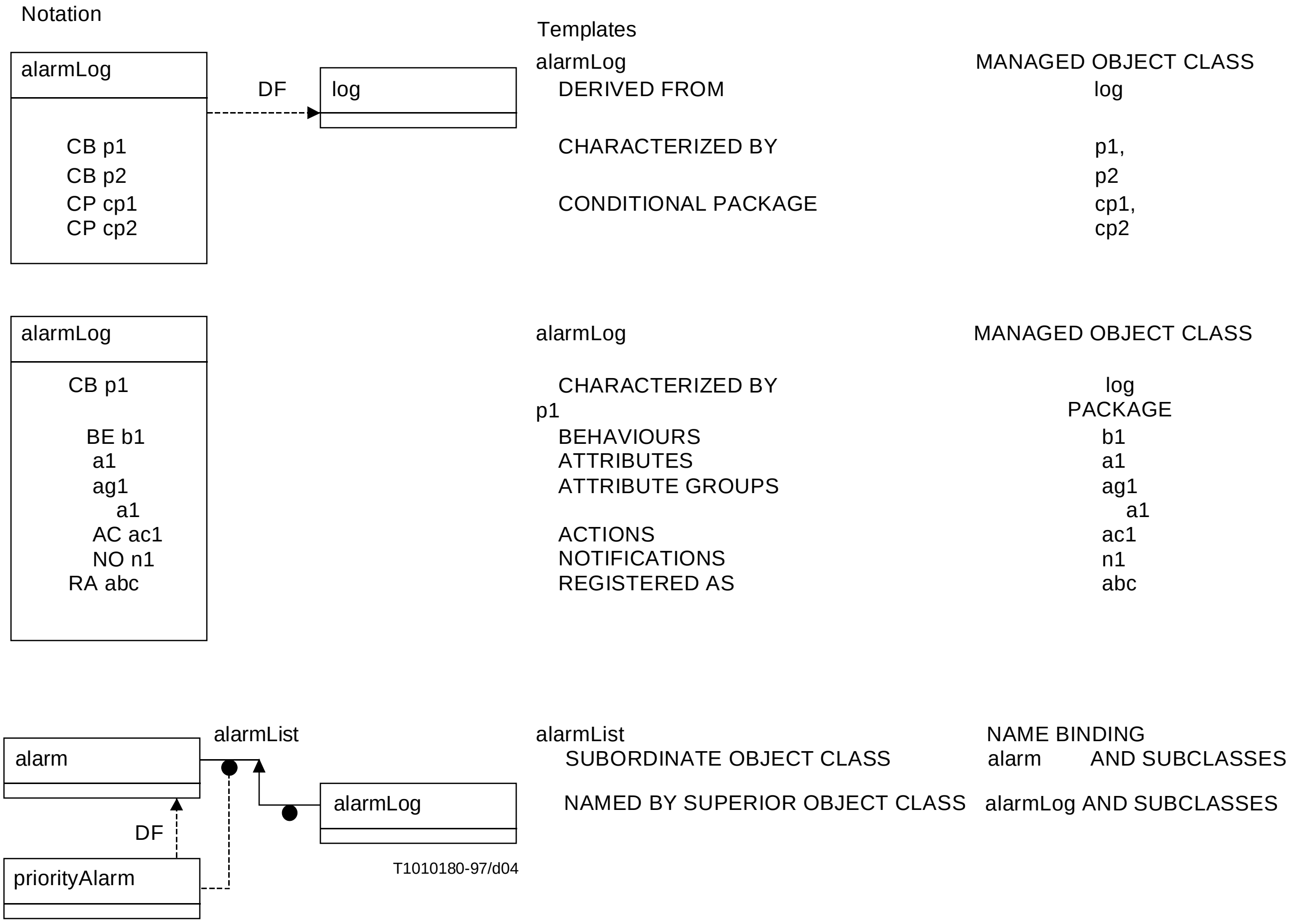
```
alarmList                              NAME BINDING
    SUBORDINATE OBJECT CLASS               alarm
    NAMED BY SUPERIOR OBJECT CLASS     alarmLog
```

```
alarmList                              NAME BINDING
    SUBORDINATE OBJECT CLASS               alarm      AND SUBCLASSES
    NAMED BY SUPERIOR OBJECT CLASS     alarmLog   AND SUBCLASSES
```

```
alarmList                              NAME BINDING
    SUBORDINATE OBJECT CLASS               alarm      AND SUBCLASSES
    NAMED BY SUPERIOR OBJECT CLASS     alarmLog   AND SUBCLASSES
```

Dashed lines are optionally provided to point out the subclasses.

Figure 3/Z.360 – Connection symbols



Templates

```
alarmLog                                  MANAGED OBJECT CLASS
   DERIVED FROM                                    log

   CHARACTERIZED BY                                p1,
                                                   p2
   CONDITIONAL PACKAGE                             cp1,
                                                   cp2
```

```
alarmLog                                  MANAGED OBJECT CLASS

   CHARACTERIZED BY                                 log
p1                                                PACKAGE
   BEHAVIOURS                                       b1
   ATTRIBUTES                                       a1
   ATTRIBUTE GROUPS                                 ag1
                                                      a1
   ACTIONS                                          ac1
   NOTIFICATIONS                                    n1
   REGISTERED AS                                    abc
```

```
alarmList                                   NAME BINDING
    SUBORDINATE OBJECT CLASS                alarm       AND SUBCLASSES

    NAMED BY SUPERIOR OBJECT CLASS          alarmLog AND SUBCLASSES
```

Figure 4/Z.360 – Examples of shorthand and combined notations

3 Example usage of the graphic notation

The purpose of this clause is to show how the graphic notation can be used. The example shown depicts a realistic graph.

The example depicts the relevant information contained in Recommendation X.741 Objects and Attributes for Access Control. The most compact form of the graphic notation is used.

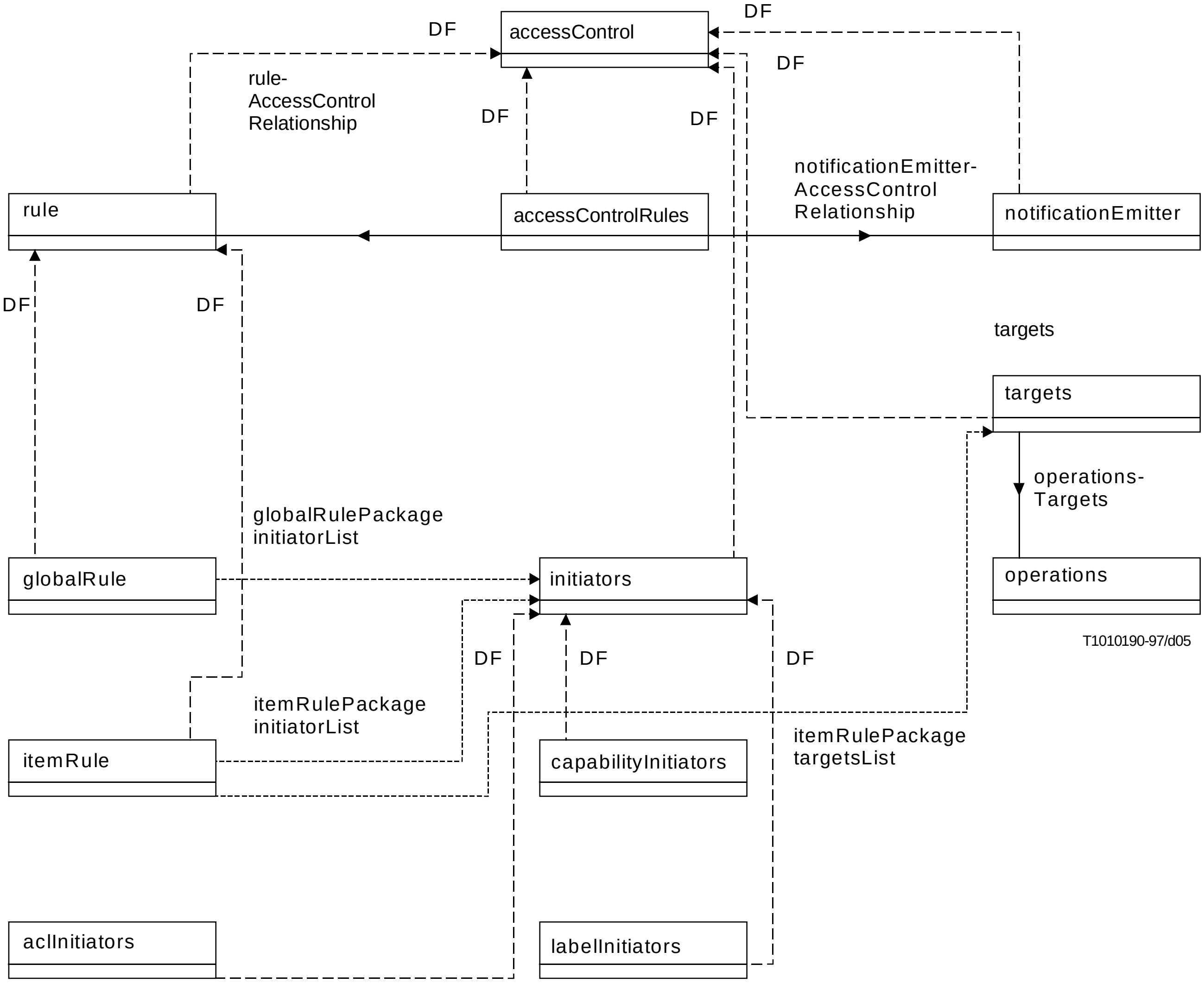


Figure 5/Z.360 – Example compact graphic notation

Figure 6 shows the same graph extended with references to packages and attributes.

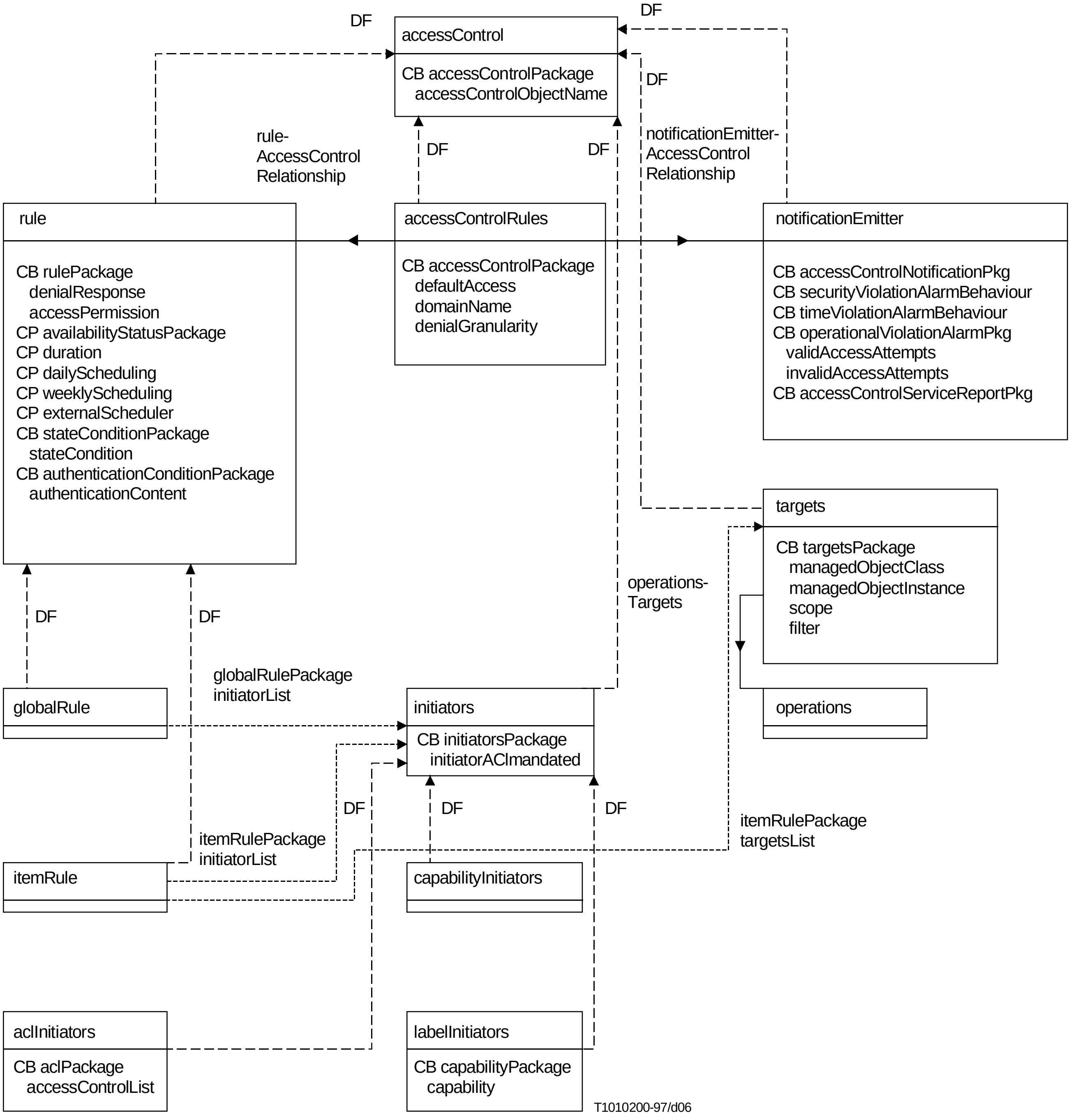


Figure 6/Z.360 – Example composed and compact graphic notation

4 Graphic notation for templates

4.1 Managed Object Class template

Figure 7 depicts a detailed graphic notation for a subset of the information provided in the Managed Object Class template. Note the use of prefix abbreviations to indicate the different kind of items.

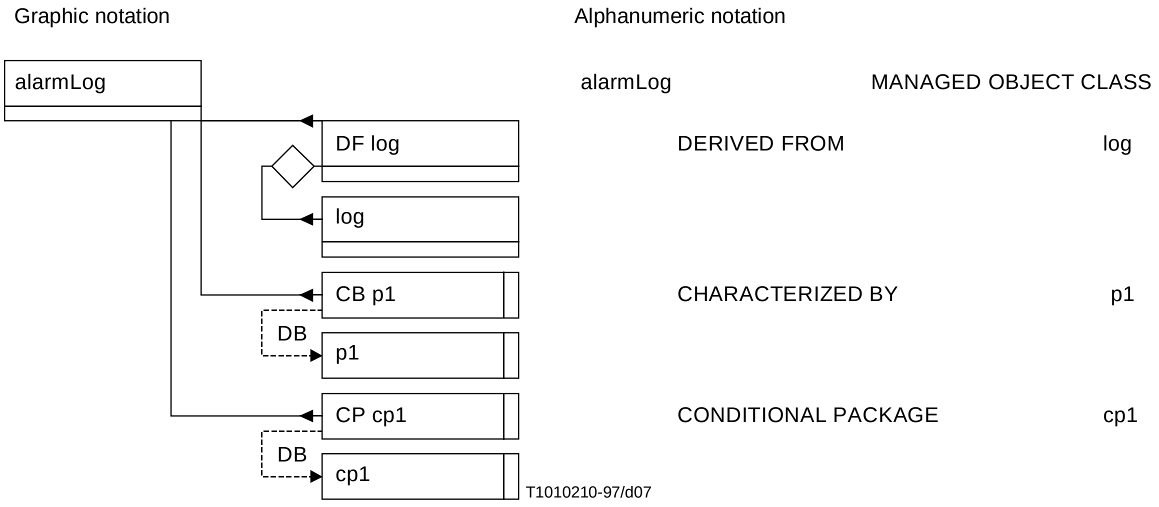


Figure 7/Z.360 – Managed Object Class template

Figure 8 shows a shorthand notation providing the same information as in Figure 7. The involved DB reference is suppressed.

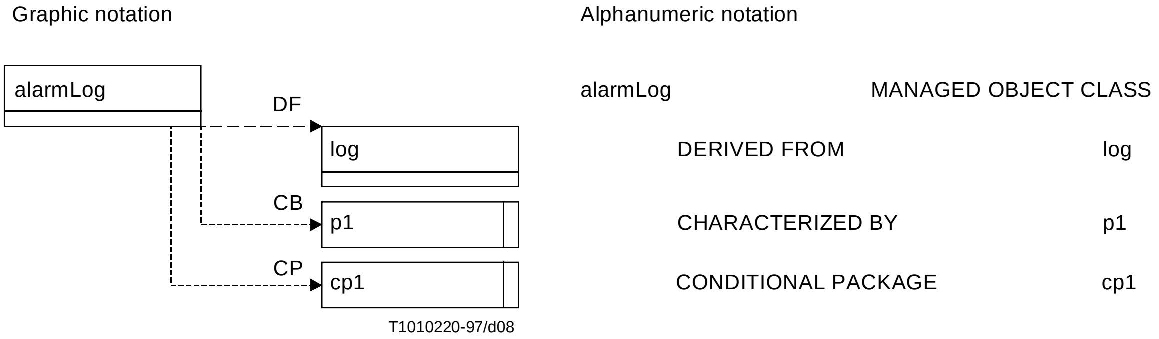
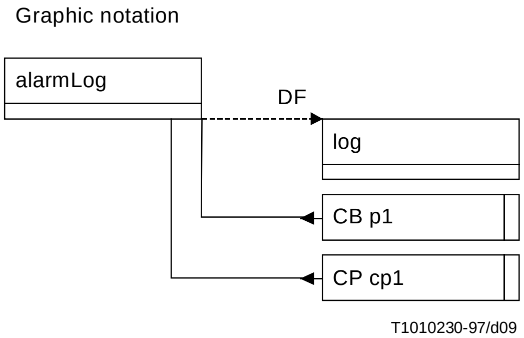


Figure 8/Z.360 – Shorthand notation

Figure 8 provides references from a Managed Object Class to a superclass (DF) and characterizing (CB) and conditional (CP) packages.

Figure 9 shows a subset of the information in Figure 7. Only the role packages contained in the Managed Object Class are shown, and the full references to Packages are discarded. Figure 8 shows more details than Figure 9. Figures 7, 8 and 9 provide explanations to the more practical notation shown in Figure 10.



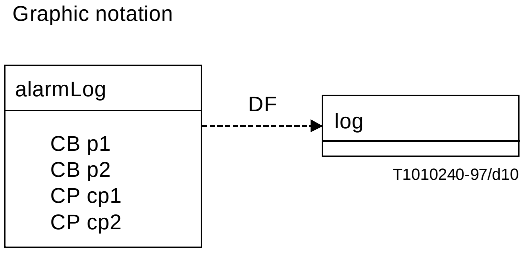
Alphanumeric notation

```
alarmLog                 MANAGED OBJECT CLASS

         DERIVED FROM                         log

         CHARACTERIZED BY                     p1

         CONDITIONAL PACKAGE                  cp1
```

Figure 9/Z.360 – Abbreviated notation

The abbreviated notation shown in Figure 9 is not capable of distinguishing inline specification of Packages from external specification of Packages.

Figure 10 shows the most compact notation. Indentations are used to depict which item is subordinate to which. The Managed Object Class label is placed in the upper left corner.



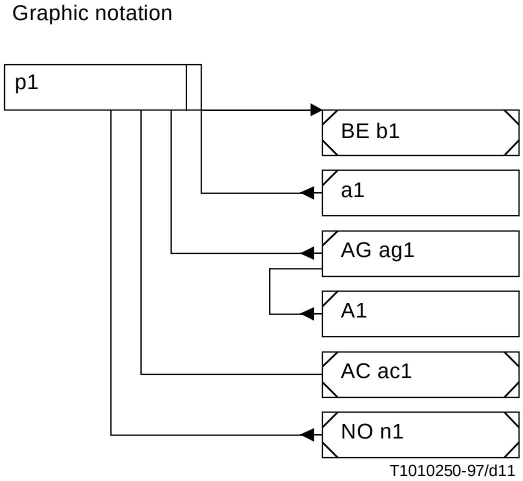
Alphanumeric notation

```
alarmLog                 MANAGED OBJECT CLASS
         DERIVED FROM                         log
         CHARACTERIZED BY                     p1,
                                              p2
         CONDITIONAL PACKAGE                  cp1,
                                              cp2
```

Figure 10/Z.360 – Compact notation

4.2 Package template

Figure 11 depicts a detailed graphic notation for a subset of the information provided in the Package template. Attribute a1 extends the definition of Attribute group ag1. See a more complete exposition in 4.4.



Alphanumeric notation

```
p1       PACKAGE

         BEHAVIOUR                  b1

         ATTRIBUTES                 a1

         ATTRIBUTE GROUPS           ag1

                                       a1

         ACTIONS                    ac1

         NOTIFICATIONS              n1
```

Figure 11/Z.360 – Package template

Figure 12 shows a compact notation for Package templates. Indentation is used to indicate which item is subordinate to which. Attributes have no prefix.

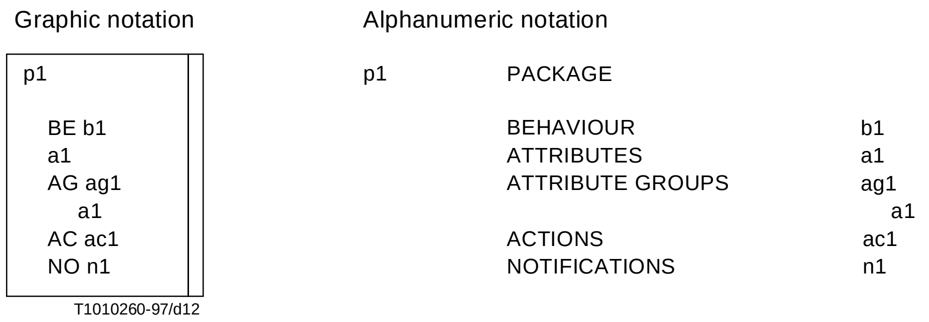


Figure 12/Z.360 – Compact notation

Managed Object Class templates and information in referenced Package templates can be depicted as if the package information is inherited into the Managed Object Class template.

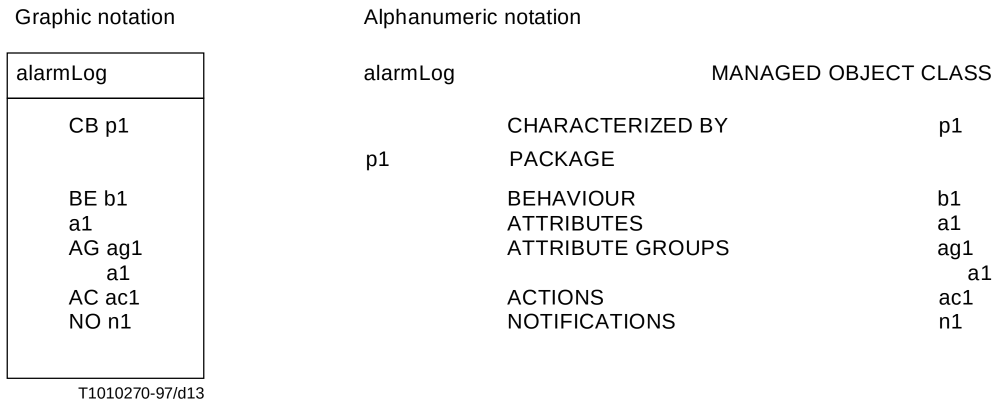


Figure 13/Z.360 – Combined notation

4.3 Name Binding template

The Name Binding template states that instances of the subordinate Managed Object Class are identified locally to an instance of the superior Managed Object Class. Name Binding is depicted by an arrowhead in the middle of the connecting line, pointing to the superior class.

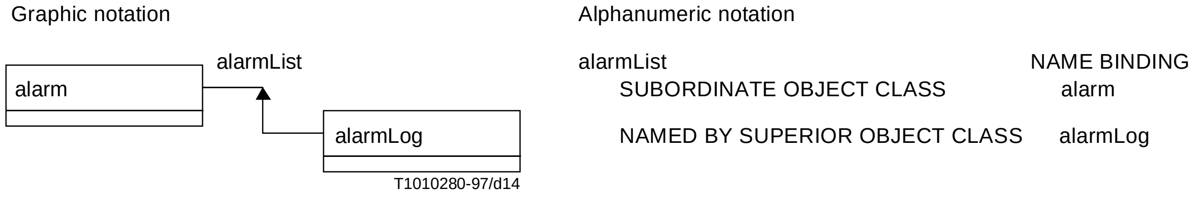
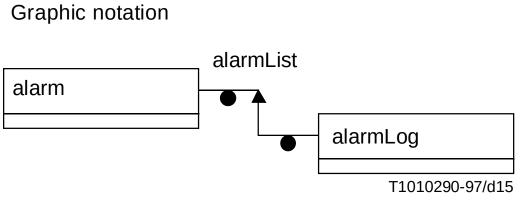


Figure 14/Z.360 – Name Binding template

The And Subclasses sub-statements are indicated by the dots on the Name Binding lines, as shown in Figure 15.

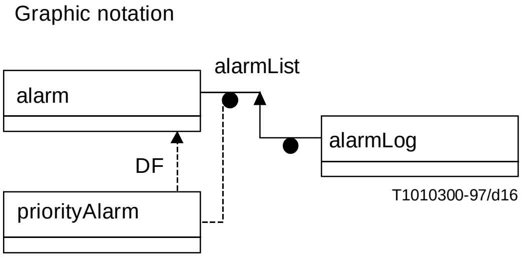


Alphanumeric notation

```
alarmList                                    NAME BINDING
    SUBORDINATE OBJECT CLASS              alarm      AND SUBCLASSES

    NAMED BY SUPERIOR OBJECT CLASS    alarmLog   AND SUBCLASSES
```

Figure 15/Z.360 – Example And Subclasses

Dashed lines are optionally provided to point out the subclasses for which the Name Binding applies.



Alphanumeric notation

```
alarmList                                    NAME BINDING
    SUBORDINATE OBJECT CLASS              alarm      AND SUBCLASSES

    NAMED BY SUPERIOR OBJECT CLASS    alarmLog   AND SUBCLASSES
```

Figure 16/Z.360 – Indication of subclasses for which the Name Binding applies

4.4 Connections

Figure 17 depicts connections between a few example symbols. The corresponding alphanumeric templates are indicated above the graph. ASN.1 specifications and further detailing of the templates are not shown. The reader of the alphanumeric specifications will have to look up each of the indicated templates, while the graphic notation can depict all the referenced items in one graph.

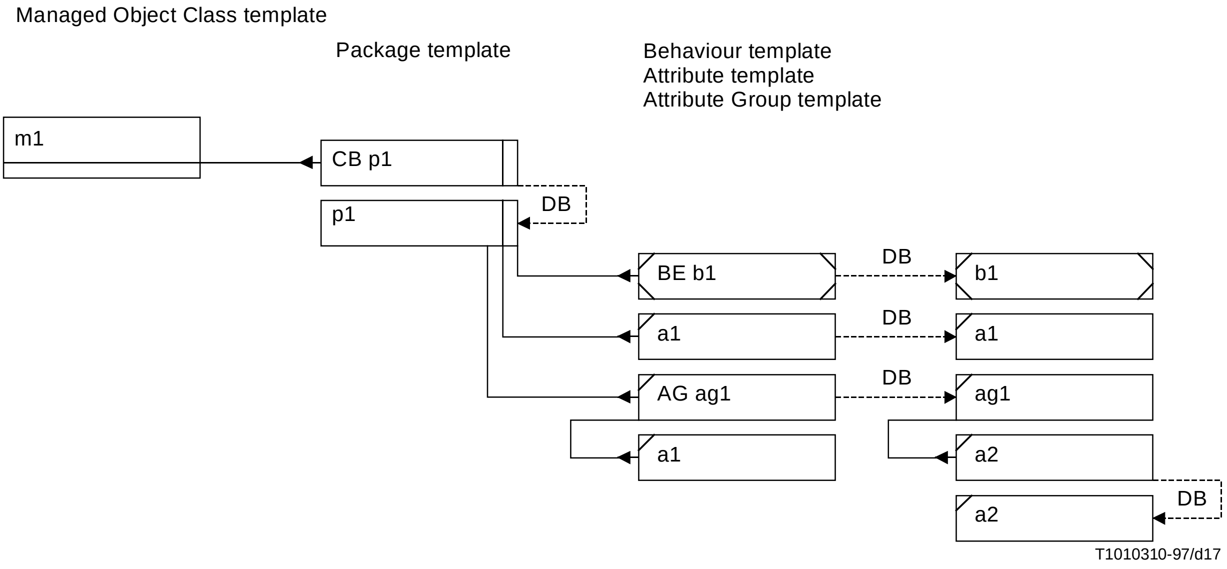


Figure 17/Z.360 – Depiction of some detailed references from an alphanumeric specification

The above example shows a statement for a managed object class m1, which contains a sub-statement CHARACTERIZED BY p1. p1 is defined by the package statement labelled p1. The package statement p1 contains a sub-statement BEHAVIOUR b1, which behaviour is defined by the behaviour package b1. The package statement p1 also contains a sub-statement ATTRIBUTES a1, which attribute is defined by the attribute statement a1. The package statement p1 also contains a sub-statement ATTRIBUTE GROUPS ag1, which attribute group is defined by the attribute group statement ag1. The attribute group sub-statement ag1 contains a sub-statement ATTRIBUTES a1 (which extends the definition of the attribute group within this package). The attribute a1 is also defined by the attribute statement a1. The attribute group statement for ag1 contains a sub-statement ATTRIBUTES a2, which attribute is defined by the attribute statement a2. Note that we have not shown the complete managed object class statement. There is a requirement that the attribute a2 in attribute group ag1 be referenced in the ATTRIBUTES sub-statement in one or more of the packages referred to in the managed object class definition. This is not shown in the above example.

5 References by attributes

This clause provides a graphic notation for references stated in informal text associated with attributes in TMN specifications.

One-way references are depicted as shown in Figure 18.

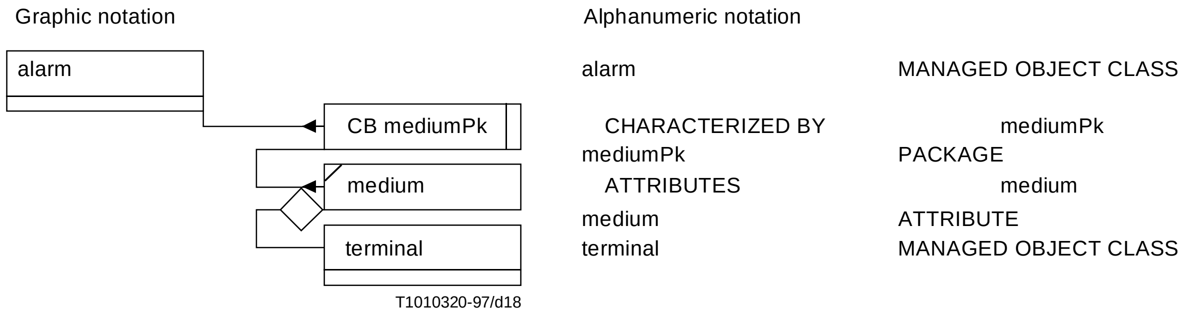


Figure 18/Z.360 – One-way reference

In Figure 18 the medium attribute contains a reference in the informal text to the terminal Managed Object Class. The value of the attribute is referring to the value of the distinguished name of an instance of the terminal Managed Object Class.

Figure 19 shows an abbreviated notation for the information in Figure 18. The package label appears as a part of the role label.

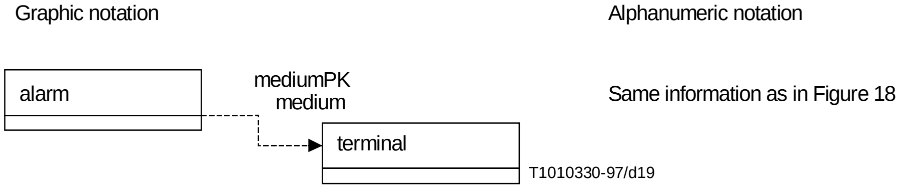


Figure 19/Z.360 – Abbreviated notation

Two mutually dependent one-way references are depicted as a two-way arrow as shown in Figure 20.

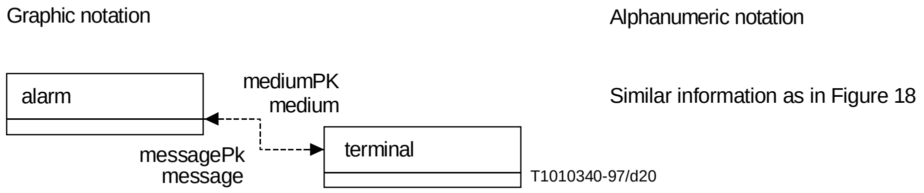


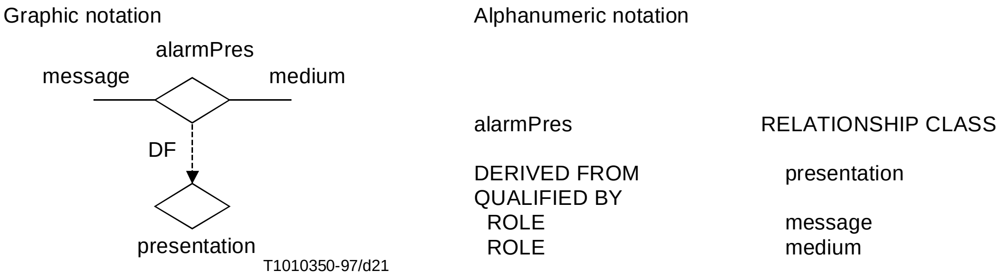
Figure 20/Z.360 – Two-way reference

The role labels should be placed close to the corresponding arrowheads.

6 Relationship Class template

This clause provides a graphic notation for a subset of the information provided in the templates defined in Recommendation X.725, Information technology – Open Systems Interconnection – Structure of management information: General Relationship Model. This is an extension of the notation introduced in the previous clauses.

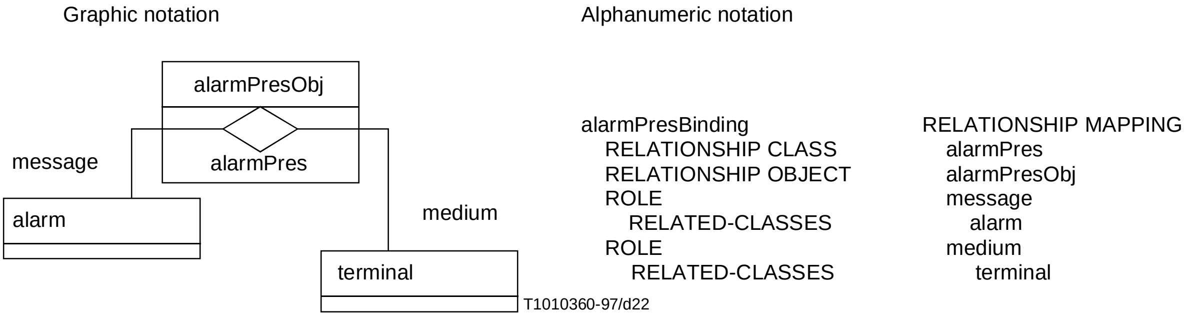
Figure 21 shows how Relationship Classes and inheritance of Relationship Classes are depicted. Relationship Classes are depicted using diamond symbols.



```
alarmPres              RELATIONSHIP CLASS

DERIVED FROM             presentation
QUALIFIED BY
  ROLE                   message
  ROLE                   medium
```

Figure 21/Z.360 – Relationship Class templace

Figure 22 depicts a Relationship Object Class (rectangle) and Role Classes (along the lines). Note that the Role Binding label is not shown in the graph, while the Relationship Class label always appears and the Relationship Object Class label appears if the rectangle appears.



```
alarmPresBinding               RELATIONSHIP MAPPING
   RELATIONSHIP CLASS             alarmPres
   RELATIONSHIP OBJECT            alarmPresObj
   ROLE                           message
      RELATED-CLASSES                alarm
   ROLE                           medium
      RELATED-CLASSES                 terminal
```

Figure 22/Z.360 – Relationship Mapping template

7 Long labels

Sometimes references have to be made to items of other Recommendations. This can be done by including the item of the other Recommendation into the graph, much like that of the alphanumeric specifications and shown in Figure 23.

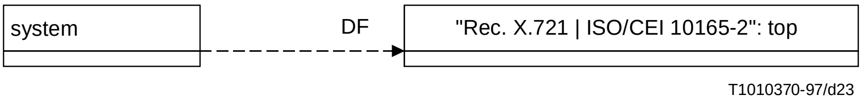
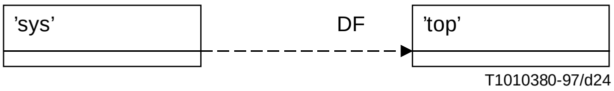


Figure 23/Z.360 – Example reference to another Recommendation

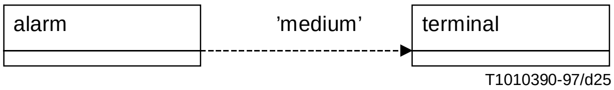
Some labels can be too long to be conveniently placed in the graph. Then they can be placed in separate columns and referred to by single quotation marks, as shown in Figure 24. This is an extension of the notation introduced in the previous clauses.



'sys'	system
'top'	"Rec. X.721 | ISO/IEC 10165-2": top

Figure 24/Z.360 – Example treatment of long labels

Long labels for attributes, attribute groups, for references by attributes, relationships and other can be referred to in the same way as for Managed Object Classes.



'medium'	mediumPk, medium

Figure 25/Z.360 – Example short label for the full label in Figure 19

Appendix I

Graphic GDMO Instantiation

Instantiation of GDMO specification can, in principle, be carried out in three stages, which do not have to be followed in a strict sequence:

1) inheritance;

2) translation;

3) instantiation.

Figure I.1 depicts a GDMO specification using Graphic GDMO.

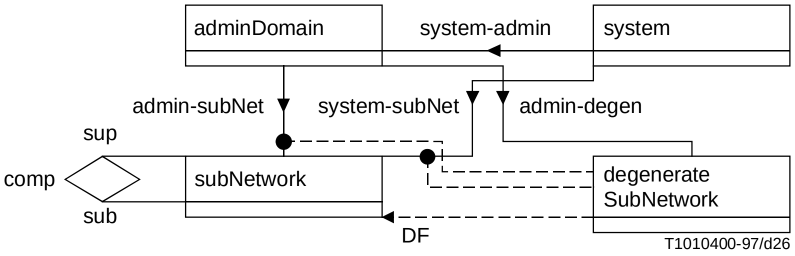


Figure I.1/Z.360 – Example GDMO specification

Inheritance stage: The inheritance stage executes Derived From statements of Managed Object Classes and Relationship Classes, Characterized By and Conditional Packages references – and implicitly Defined By statements of the metalanguage – Package statements, Attributes references and statements, Attribute Groups references and statements, and And Subclasses statement of Name Bindings. This way, the properties of the object classes are no more referred to, they are inherited into the object class to be instantiated.

Figure I.2 shows the same specification as shown in Figure I.1 after the inheritance stage has been completed. Note the inherited Name Bindings and Relationship Classes.

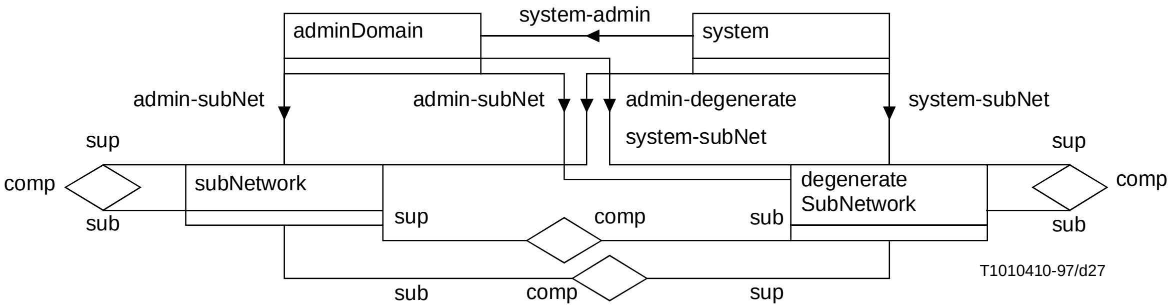


Figure I.2/Z.360 – Example GDMO specification after completion of the inheritance stage

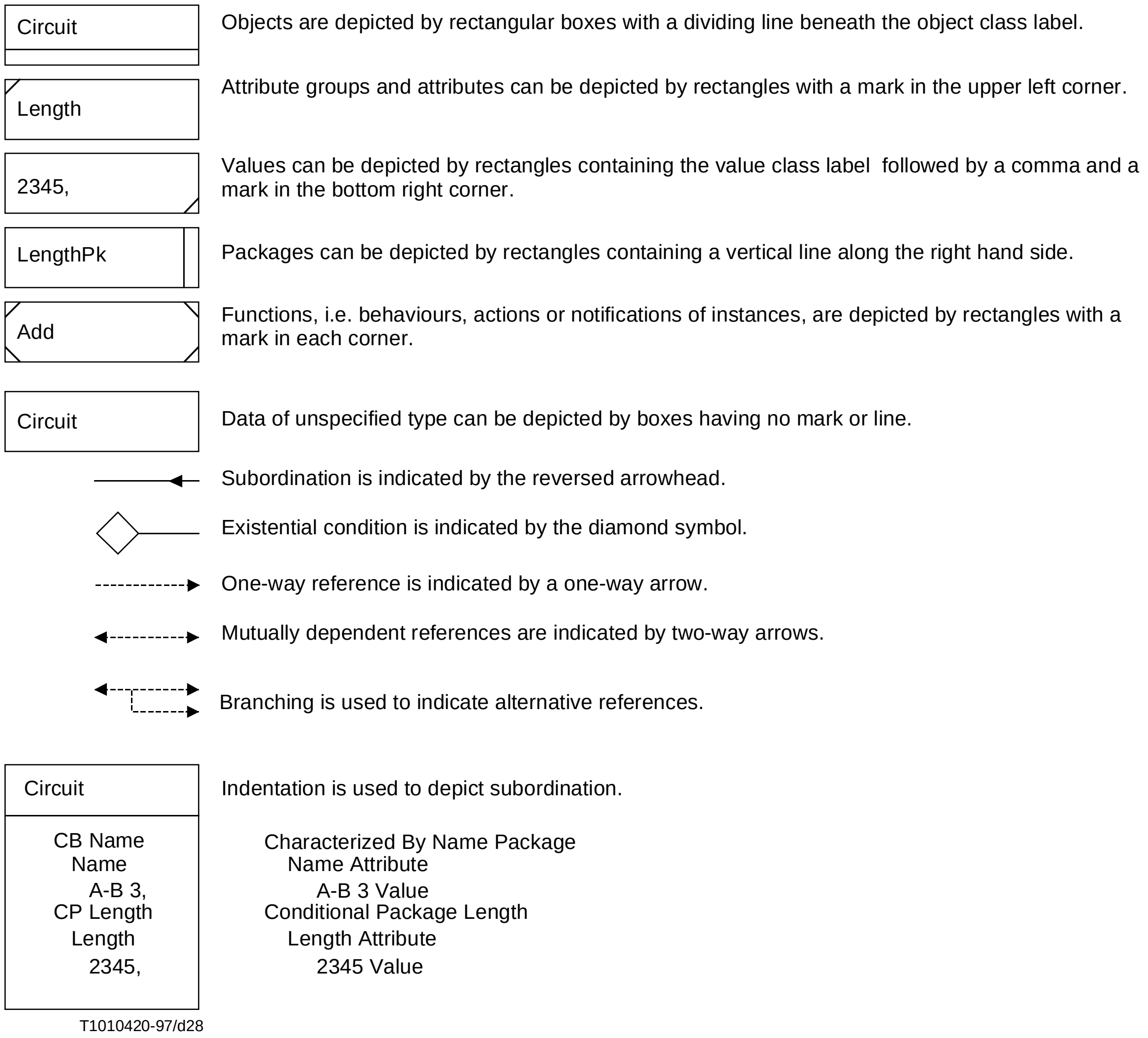
The user of the GDMO instantiation technique is free to depict the items of interest only.

Translation stage: While the Name Binding statements make up a network between globally defined classes, instances form trees. Therefore, subordination (by relative distinguished name) of object instances is a different language notion from Subordinate Object Class of Name Binding. Hence, two somewhat different languages are needed for classes and instances. The graphic notation used for instances of GDMO specifications is shown in Figure I.3.

The Graphic instantiation language is similar to Graphic GDMO, however, the instantiation language does not contain Name Binding and Derived From statements.

A standard set of translations from the Graphic GDMO for classes to the Graphic instantiation language is needed. The Graphic instantiation language is applicable both for instances and classes.

Figure I.4 shows the result after the GDMO specification in Figure I.2 has been translated into a specification using the Graphic instantiation language. Note that adminDomain can, as shown, either be identified globally or locally to system. Note also the use of two-way references with branching.



Objects are depicted by rectangular boxes with a dividing line beneath the object class label.

Attribute groups and attributes can be depicted by rectangles with a mark in the upper left corner.

Values can be depicted by rectangles containing the value class label followed by a comma and a mark in the bottom right corner.

Packages can be depicted by rectangles containing a vertical line along the right hand side.

Functions, i.e. behaviours, actions or notifications of instances, are depicted by rectangles with a mark in each corner.

Data of unspecified type can be depicted by boxes having no mark or line.

Subordination is indicated by the reversed arrowhead.

Existential condition is indicated by the diamond symbol.

One-way reference is indicated by a one-way arrow.

Mutually dependent references are indicated by two-way arrows.

Branching is used to indicate alternative references.

Indentation is used to depict subordination.

Characterized By Name Package
   Name Attribute
      A-B 3 Value
Conditional Package Length
   Length Attribute
      2345 Value

Figure I.3/Z.360 – Graphic instantiation language

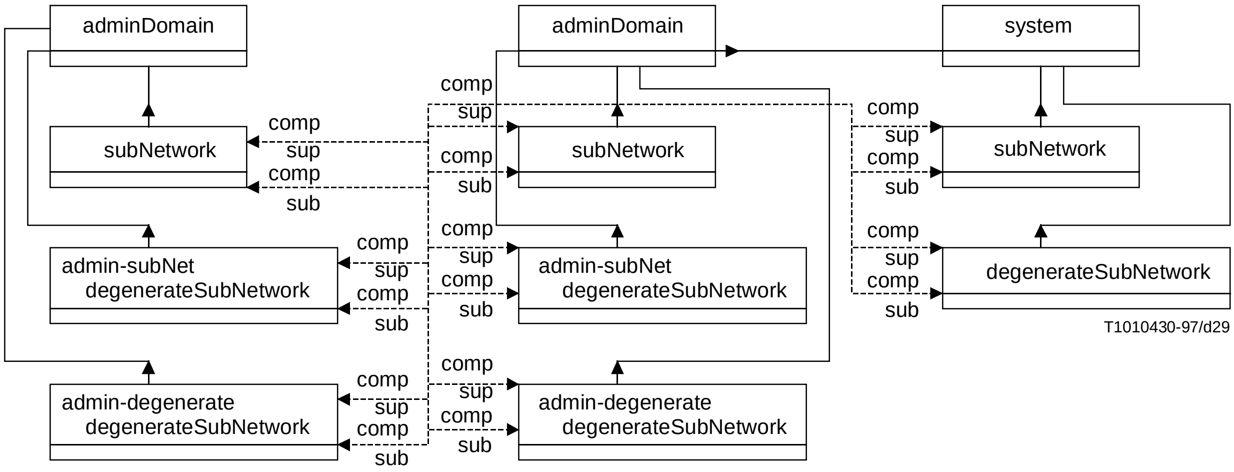


Figure I.4/Z.360 – Example classes prepared for possible instantiation

Name Binding is replaced by subordination; the subordinate object class is labelled by the concatenation of the Name Binding label and the label of the subordinate Managed Object Class. The concatenation is indicated by indentation of the Subordinate Managed Object Class label. Note that due to inheritance by ’And Subclasses’, there can be several alternative Name Bindings between two GDMO classes. Each alternative Name Binding can result in a separate subordination branch. If only one Name Binding is possible, the Name Binding label can be left out in the label of the subordinate object class, however, the indentation cannot.

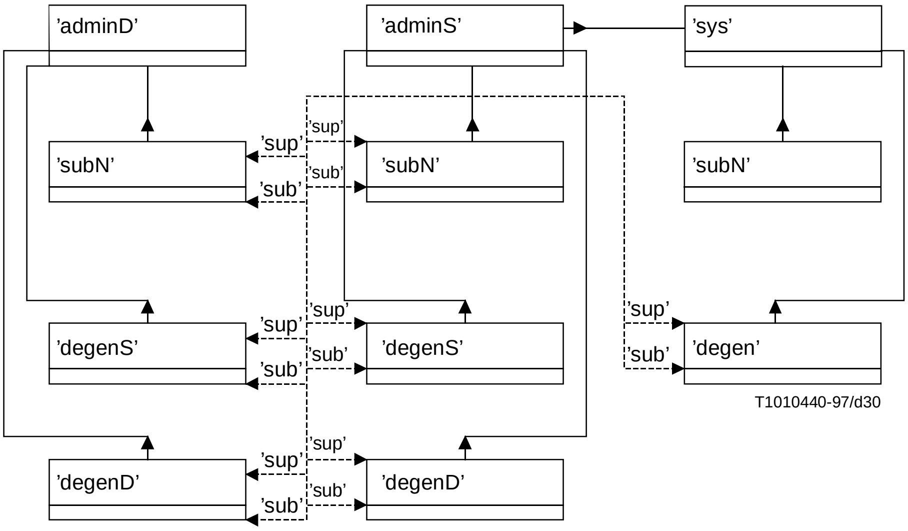
Relationship classes are normally replaced by two-way mutually dependent references. The roles are labelled by the concatenation of the Relationship Class label and Role label.

If Relationship Objects are defined in a Role Binding template, the two-way references are made to the Relationship Objects, which are treated as ordinary objects. The roles pointing to the Relationship Object are blank, while the roles pointing to the connected Managed Object Classes are equal to the Role labels only – without concatenation with the Relationship Class label. This simplification is possible because the same role label can appear only once as seen from a Relationship Object towards a Managed Object Class, however, this is not always the case for roles between two Managed Object Classes.

All other labels are unchanged in the translation process.

Note that Figure I.4 still represents classes, however, now in a notation which is suited for instantiation.

As shown in Figure I.4, the full labels can become very long. Therefore, short labels within single quotation marks can be used, as shown in Figure I.5.



’sys’	system
’adminD’	adminDomain
’adminS’	system-admin adminDomain
’subN’	admin-subNet subNetwork
’degen’	system-subNet degenerateSubNetwork
’degenS’	admin-subNet degenerateSubNetwork
’degenD’	admin-degen degenerateSubNetwork
’sup’	comp, sup
’sub’	comp, sub

Figure I.5/Z.360 – Example classes for instantiation

The graphic instantiation language provides more information than what can be found in the distinguished name of the instances.

Instantiation stage: Instantiation of the classes follows the translation stage. Each instance corresponds to a class, and the schema data are homomorphic to their population data.

Figure I.6 depicts some instances according to a subset of the classes shown in Figure I.5. As instance graphs typically depict many items, there will frequently not be available space for long labels, and quotation marks will become increasingly more useful. Note that instance labels can be shown, e.g. values of identifier attributes, but are not required in instance graphs.

Note also that references between instances are actual, hence, branching to alternatives makes no sense for instances. Therefore, two-way references without branching are used in Figure I.6.

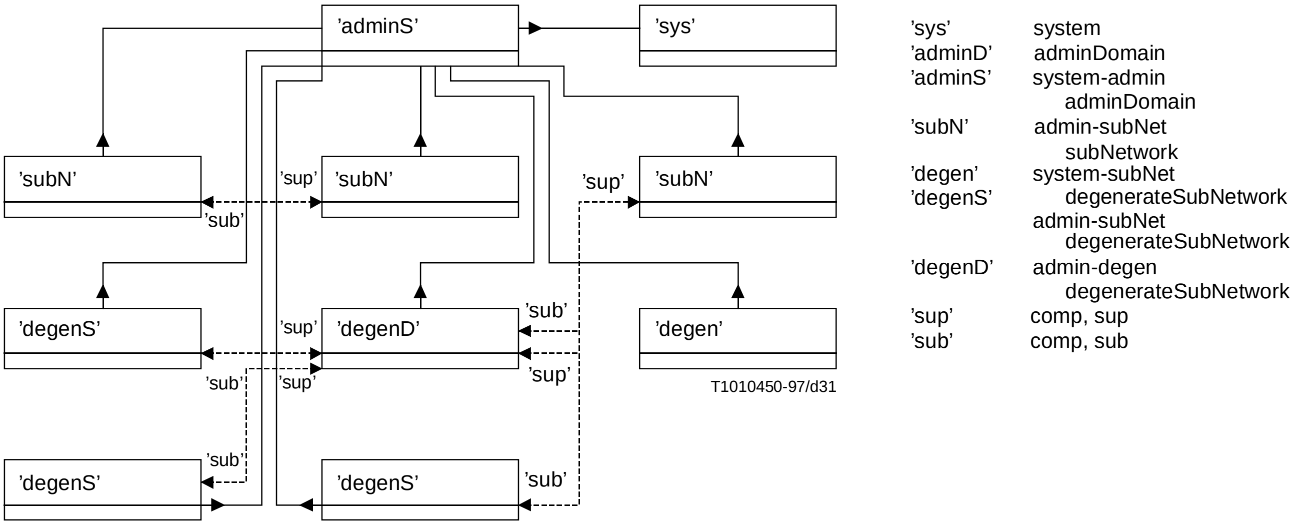


Figure I.6/Z.360 – Example instances

Since the behaviour has to be 'inherited' by the instances, the packages have to be instantiated as well. See Figure I.7.

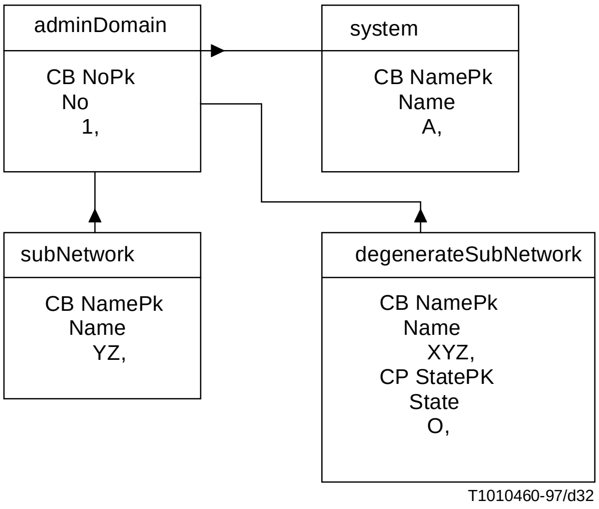


Figure I.7/Z.360 – Example instances with attributes

Appendix II

Requirements on Graphic GDMO

II.1 Background

GDMO, Guidelines for the Definition of Managed Objects, is a formalism defined in Recommendation X.722 to assist the definition of managed objects in TMN (Telecommunications Management Networks). Attributes for representing relationships are defined in Recommendation X.732. The GDMO formalism is extended with a general relationship notation in Recommendation X.725 (General Relationship Model). The overall object structure is defined by GDMO/GRM, while the syntax of the data is defined using ASN.1, Recommendations X.208 or X.680, Abstract Syntax Notation One.

The three languages are widely used by many Study Groups in ITU-T for the specification of managed objects for many application areas. The languages are purely alphanumeric. Recommendation Z.360 is developed to provide an overview of GDMO/GRM specifications. For this development, a set of requirements were developed. The provided Graphic GDMO meets all requirements, except some of the options in II.7 of the requirements.

II.2 Use of Graphic GDMO/GRM

Graphic GDMO/GRM, Graphic GDMO for short, is expected to be used by users having different needs. A unified list of requirements must take into account the different needs of these users and support cooperation between the different users.

An inexhaustive list of potential users of Graphic GDMO is given in the following:

- TMN information designers requiring an overview of existing specifications and alternative new data designs.
- TMN service designers requiring an overview of existing management information models for which and with which to provide new services.
- TMN interface designers requiring an overview of the information to be provided across the interfaces between TMNs, to and from OSFs, and out of TMN, e.g. across the TMN boundary.
- TMN implementors requiring an overview of the specifications to be implemented as data and procedures and by communication interfaces and protocols.
- Human-Machine Interface (HMI) designers requiring an overview of existing TMN specifications to be supported by an HMI, for which HMI data have to be designed and to which HMI data have to be mapped.
- Purchasers and vendors of TMN systems requiring an overview of the TMN information which is required and must be provided by the TMN system ordered.

Graphic GDMO can be used in various ways in conjunction with Alphanumeric GDMO/GRM:

- Graphic GDMO specifications can provide overview of alternative designs before any Alphanumeric GDMO/GRM specification is developed.
- Graphic GDMO specifications can provide overview of existing Alphanumeric GDMO/GRM specifications – which otherwise are very hard to convey; the graphics can be generated either manually or automatically.
- Once a tool for making Graphic GDMO drawings is available, users will request automatic generation of the corresponding alphanumeric templates and be requested to fill in the needed additional information.

The design of Graphic GDMO must take all these future users and usages into account.

Contributions of existing and new graphics for GDMO/GRM have been invited. However, the contributor organization should also itself evaluate the proposed graphics for this use, as some popular graphics for GDMO are already evaluated and have been found inconsistent with or misleading compared with the Alphanumeric GDMO/GRM.

The current requirements do not discuss how tool support can affect the design of Graphic GDMO and vice versa.

II.3 Requirements on a Graphic GDMO

II.3.1 Alphanumeric GDMO/GRM specifications shall have priority over graphs using Graphic GDMO

This means that if inconsistencies between Alphanumeric GDMO/GRM and Graphic GDMO specifications are discovered, the alphanumeric specification is normative.

This requirement shall go into the final Recommendation text.

II.3.2 The Graphic GDMO shall be true to the Alphanumeric GDMO/GRM

This subclause provides the requirements on the mapping between the graphic and the alphanumeric notations.

II.3.2.1 The Graphic GDMO shall introduce no notion which is not found in Alphanumeric GDMO/GRM for the definition of managed objects.

II.3.2.2 Specifications using Graphic GDMO shall not indicate any fact which is not stated in the Alphanumeric GDMO/GRM specifications for the definition of managed objects.

II.3.2.3 The symbols used in Graphic GDMO shall have a one-to-one correspondence to templates in Alphanumeric GDMO/GRM, i.e. there shall be one symbol only for one template and not one symbol for different templates. There may be templates in Alphanumeric GDMO/GRM for which there is no symbol in Graphic GDMO.

II.3.2.4 The connectives used in Graphic GDMO shall have a one-to-many mapping to templates and subtemplates in Alphanumeric GDMO/GRM. The purpose of the one-to-many mapping, as opposed to a one-to-one mapping, is to provide simpler overviews in the Graphic GDMO specifications.

II.3.2.5 One-way references in the Alphanumeric GDMO/GRM shall be preserved as one-way connectives in Graphic GDMO with the same direction as in the Alphanumeric GDMO/GRM.

II.3.2.6 The choice of symbols in Graphic GDMO shall be systematic. The symbols should indicate specialization, i.e. by adding additional details to a base symbol, when this is the case, but not indicate specialization when this is not the case.

II.3.2.7 The Graphic GDMO may provide means to state details which are not used to provide an overview; however, the overview graphics must be a systematic subset or derivation detailed graphics.

II.3.2.8 The Graphic GDMO may provide extensions to show contexts in which GDMO/GRM is used or implications of its use; see requirement II.7.

II.4 The Graphic GDMO shall minimally provide the following information

II.4.1 Managed object classes with associated class labels. In case of Graphic GDMO specifications spanning more than one document, document identifiers may have to be added.

II.4.2 Inheritance between managed object classes, provided by the Derived From subtemplate.

II.4.3 Name Binding provided by the Name Binding template, possibly without the Name Binding label.

II.4.4 Inheritance of Name Binding provided by the And Subclasses subtemplates.

II.4.5 One-way references, realized by Attributes within Packages and corresponding labels.

II.4.6 Pointer attributes which state a one-way reference to instances of alternative managed object classes.

II.4.7 GRM relationships with associated class and role labels.

II.4.8 Inheritance between GRM relationships, provided by the Derived From subtemplate.

II.4.9 Managed object classes which are GRM relationships, with the associated class labels.

II.5 The labels used in Graphic GDMO shall satisfy the following requirements

Graphs using Graphic GDMO shall be used to refer to Alphanumeric GDMO/GRM specifications. Therefore, the labels used in the graphics should accurately reflect labels used in the alphanumeric specifications. However, this may lead to graphs becoming rather overloaded. This may result in a need to make use of the facilities provided by requirement II.5.4.

II.5.1 The application specific labels used in Graphic GDMO graphs shall be identical to, and not abbreviations of, those of the corresponding Alphanumeric GDMO/GRM specifications.

II.5.2 The application specific labels used in Graphic GDMO graphs shall provide the full path to the item label, such that no context is lost.

II.5.3 The association between a label and its symbol/connective shall be unambiguous.

II.5.4 In the case that the application specific labels cannot fit into the symbol, means shall be provided to refer to complete labels in the same graph.

II.6 The Graphic GDMO specifications shall provide an overview of Alphanumeric GDMO/GRM specifications

These guidelines may go into an annex or appendix on Guidelines for how to use Graphic GDMO.

II.6.1 The Graphic GDMO specifications need not provide graphics for every statement and sub-statement in the Alphanumeric GDMO/GRM specifications.

II.6.2 A graph using Graphic GDMO need not provide an overview of all information in a specification using Alphanumeric GDMO/GRM, even if this information is expressible using Graphic GDMO.

II.6.3 The graphic overviews of an Alphanumeric GDMO/GRM specification can be split into several graphs, and there is no restriction on what information can be found in one graph.

II.6.4 References from a graph (or specification) in graphic GDMO are made by including the referenced items in the graph. Hence, there is no separate connective symbol between graphs, and connectivity is achieved by duplication of graphs.

II.6.5 A Graphic GDMO graph should be provided with a title and text indicating how this graph is delimited relative to the alphanumeric specifications to which it refers.

II.7 Graphic GDMO may provide extensions illustrating use and implications of GDMO/GRM specifications

These extensions do not relate to definitions of managed object classes using GDMO/GRM, but for the purposes itemized below. The definition of these extensions may become annexes or appendices of the Recommendation text.

II.7.1 Inclusion of references to behaviour specifications using SDL GR; Graphic GDMO should avoid using symbols used in SDL GR.

II.7.2 Graphs of containment trees of instances complying to GDMO/GRM specifications may be needed.

II.7.3 Graphs of mappings from the contents of HMI data schemata to GDMO/GRM specifications may be needed.

II.7.4 Graphs of what GDMO/GRM data are used in which portion (e.g. service, fragment, schema, library, interface, profile, ensemble, level, layer, view, function, Recommendation) of TMN specifications.

II.7.5 Inclusion of references to ASN.1 constructs and their size constraints.

ITU-T RECOMMENDATIONS SERIES

Series A	Organization of the work of the ITU-T
Series B	Means of expression: definitions, symbols, classification
Series C	General telecommunication statistics
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	TMN and network maintenance: international transmission systems, telephone circuits, telegraphy, facsimile and leased circuits
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks and open system communication
Series Z	**Programming languages**